Integration of Multiple Services using Application Programming Interface

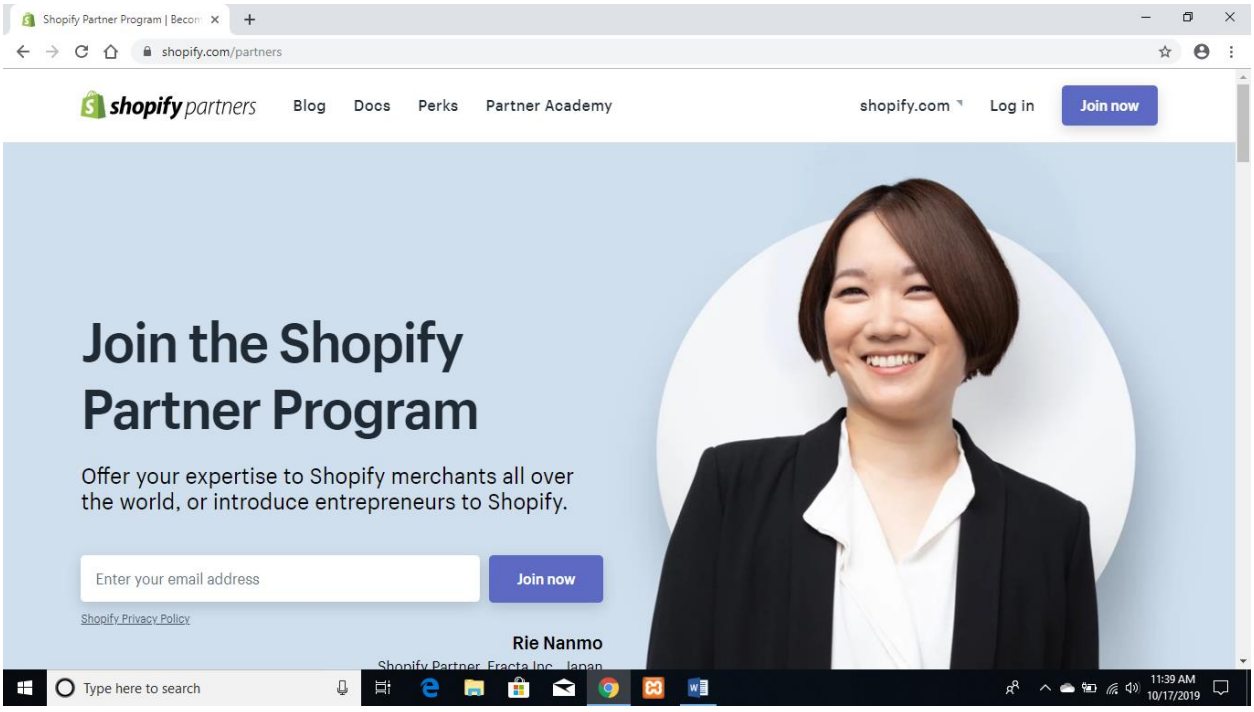
Platform: Shopify

Tools:

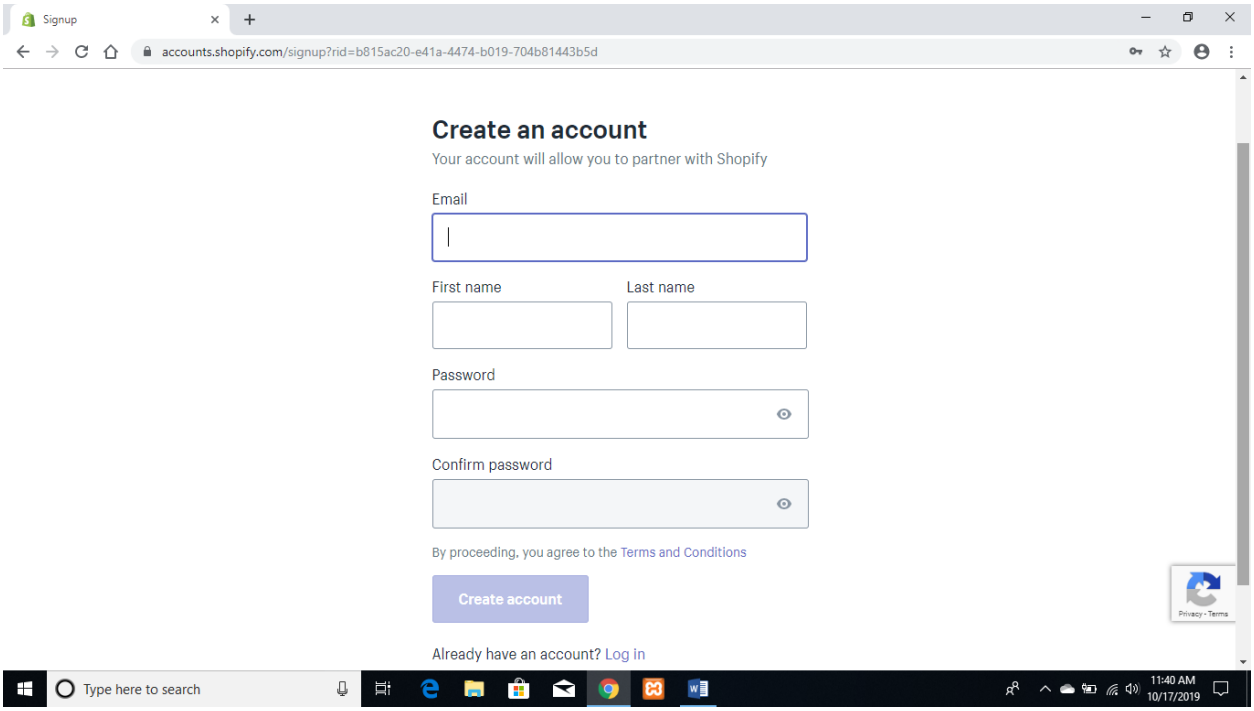
a. ngrok
b. nodejs (10.16.3 LTS)
c. Postman

Task Number 1: Creating a Shopify Partner Account

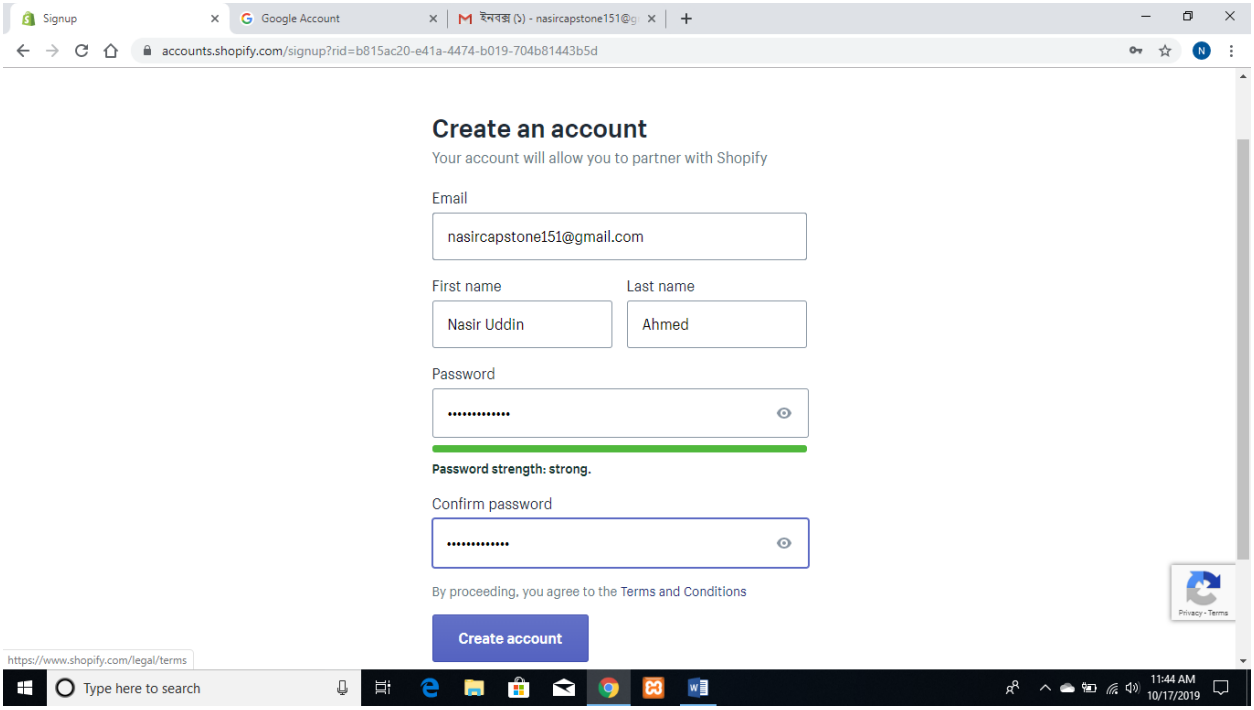
Go to https://www.shopify.com/partners



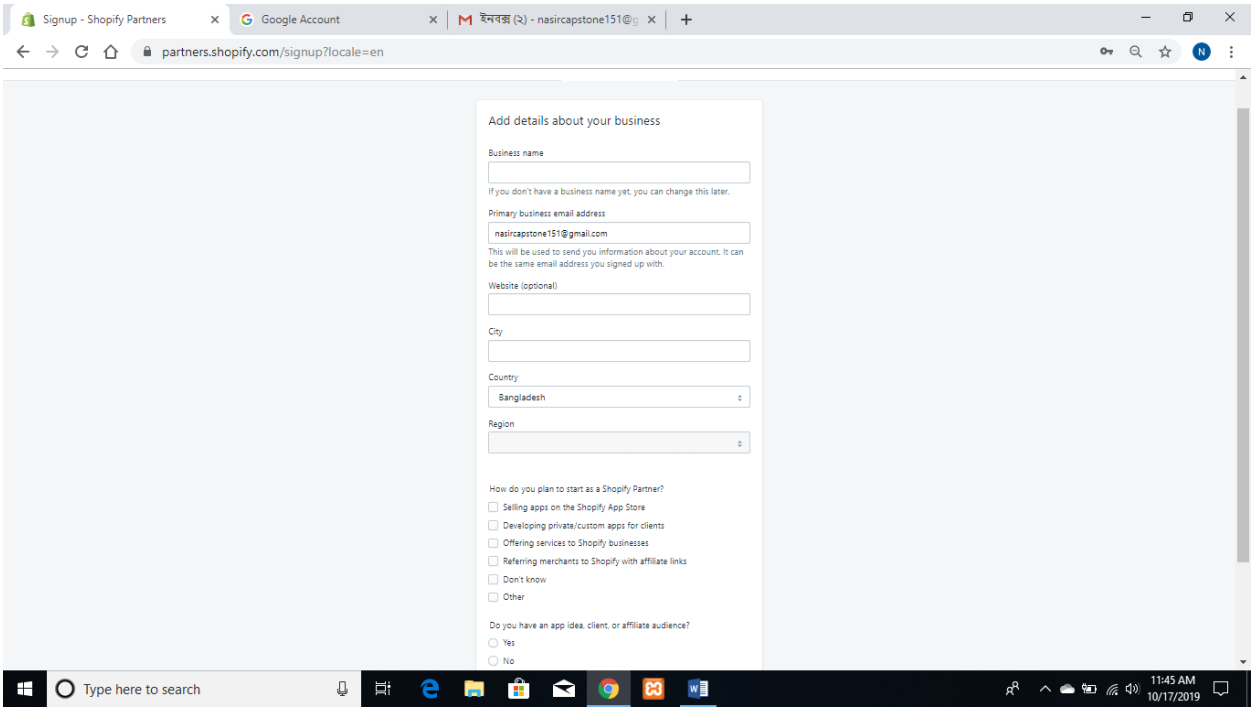
Press Join now



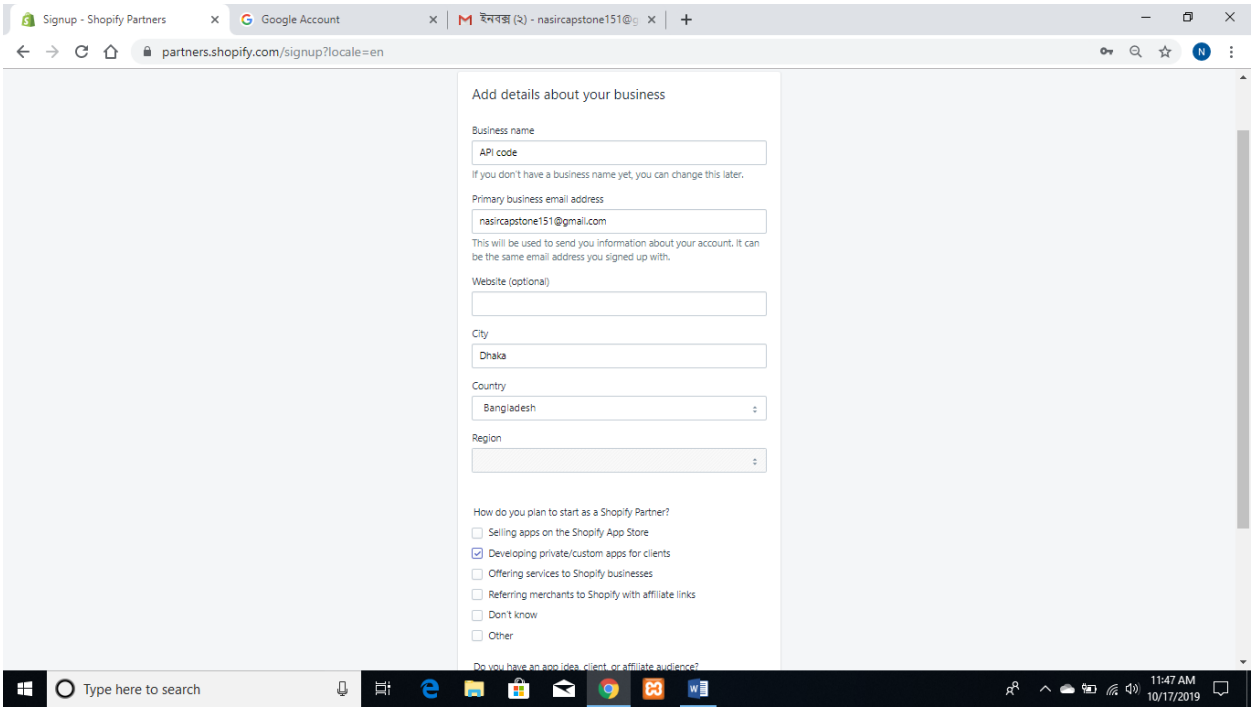
Feel up the input fields with necessary information. I have put my necessary information



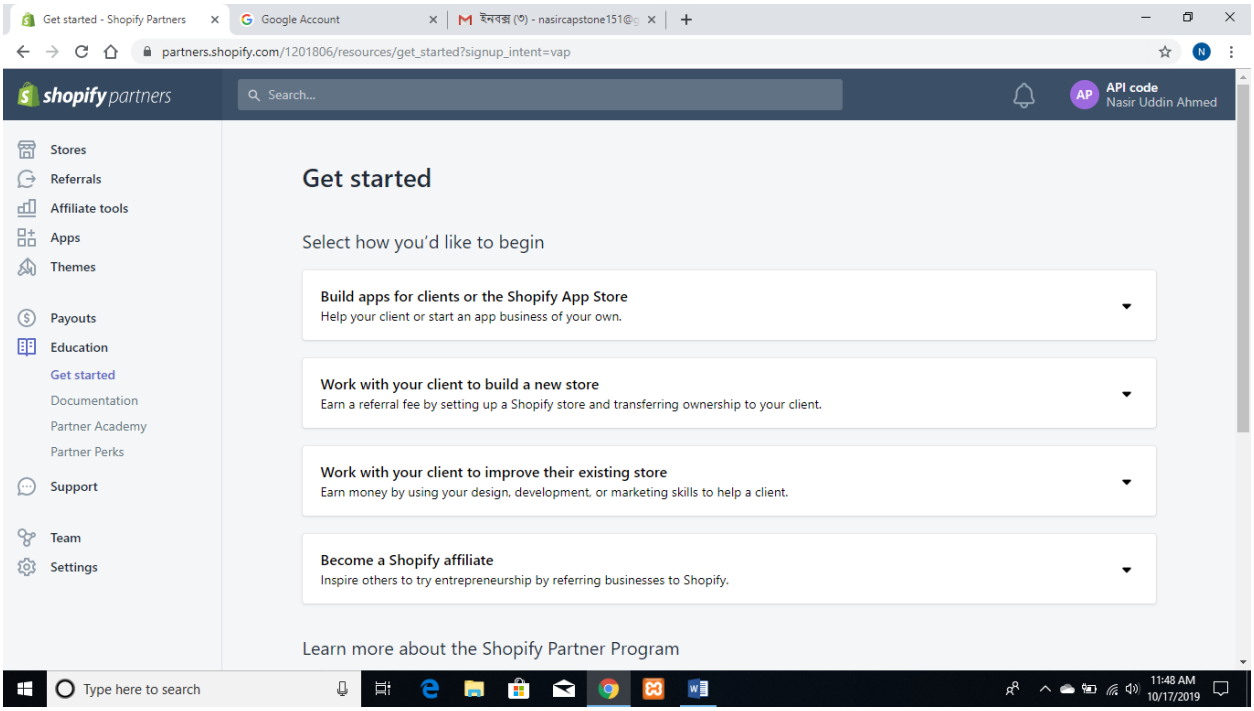
Press Create account



In this page put Business Name, City, How do you plan to start as a Shopify Partner?



Now press View your dashboard.

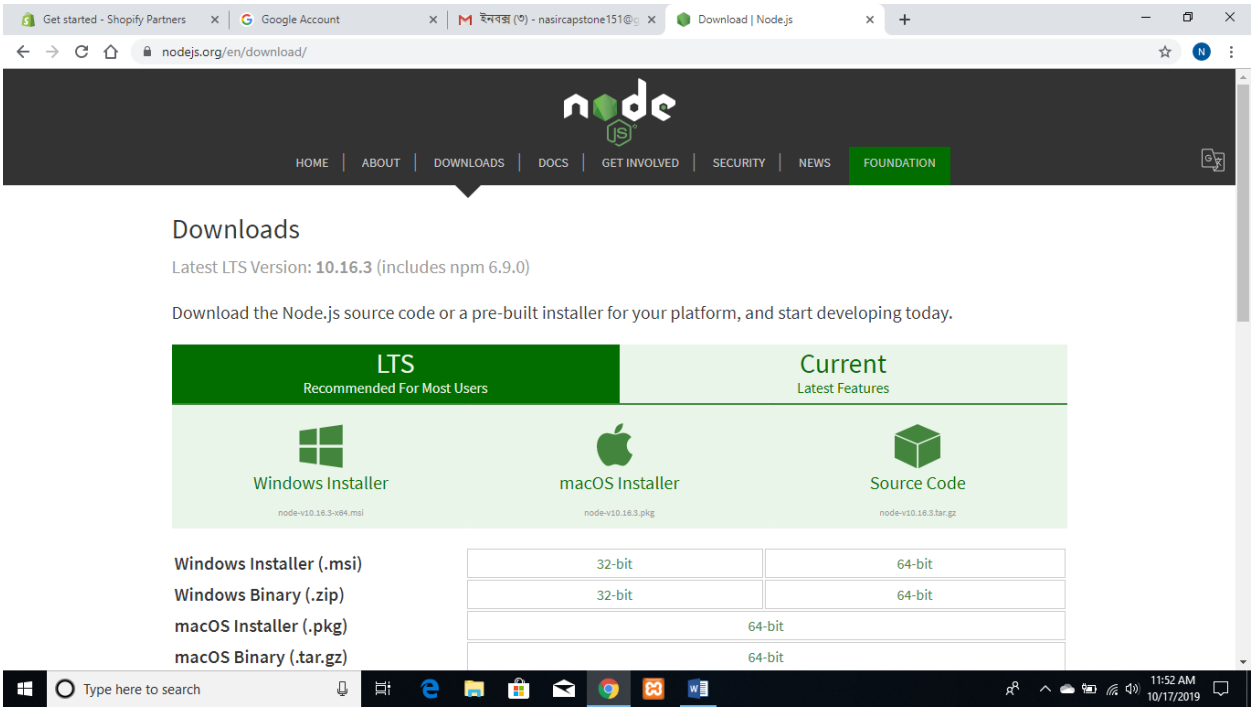


This is the Shopify(admin) dashboard.

Now we have a Shopify partner account, where we have a store. Now we will develop application.

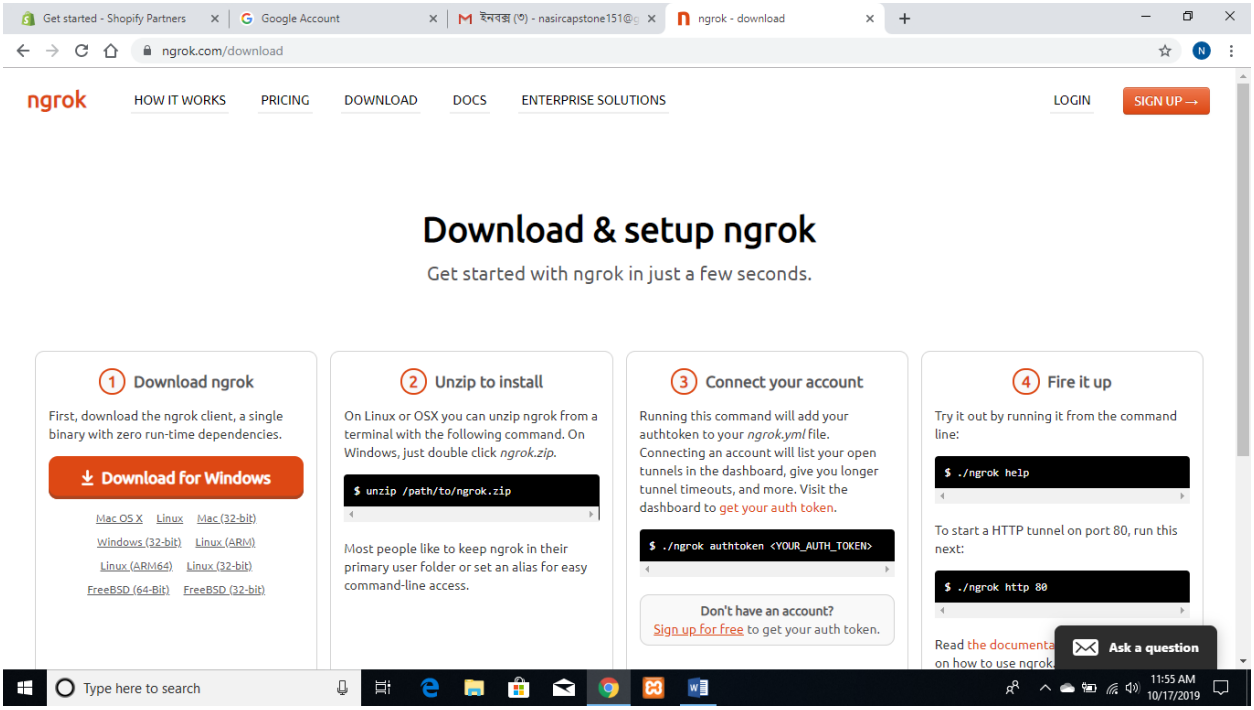
Task Number 2: Download nodejs, ngrok and postman

Go to https://nodejs.org/en/download/



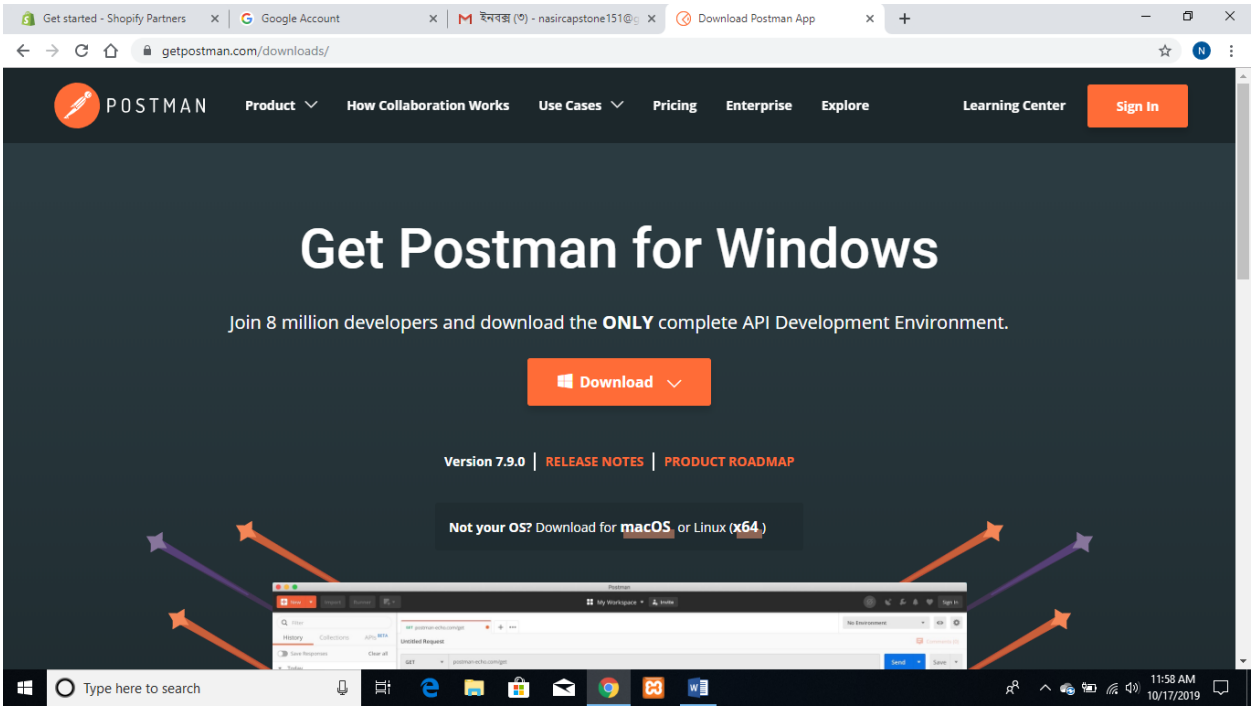
Now download Windows installer(.msi) 32/64 bit according to your computer's configuration.

Go to https://ngrok.com/download



Download according to your operating system. Here I have used windows operating, so I pressed Download for Windows.

Go to https://www.getpostman.com/downloads/



Press Download to get postman.

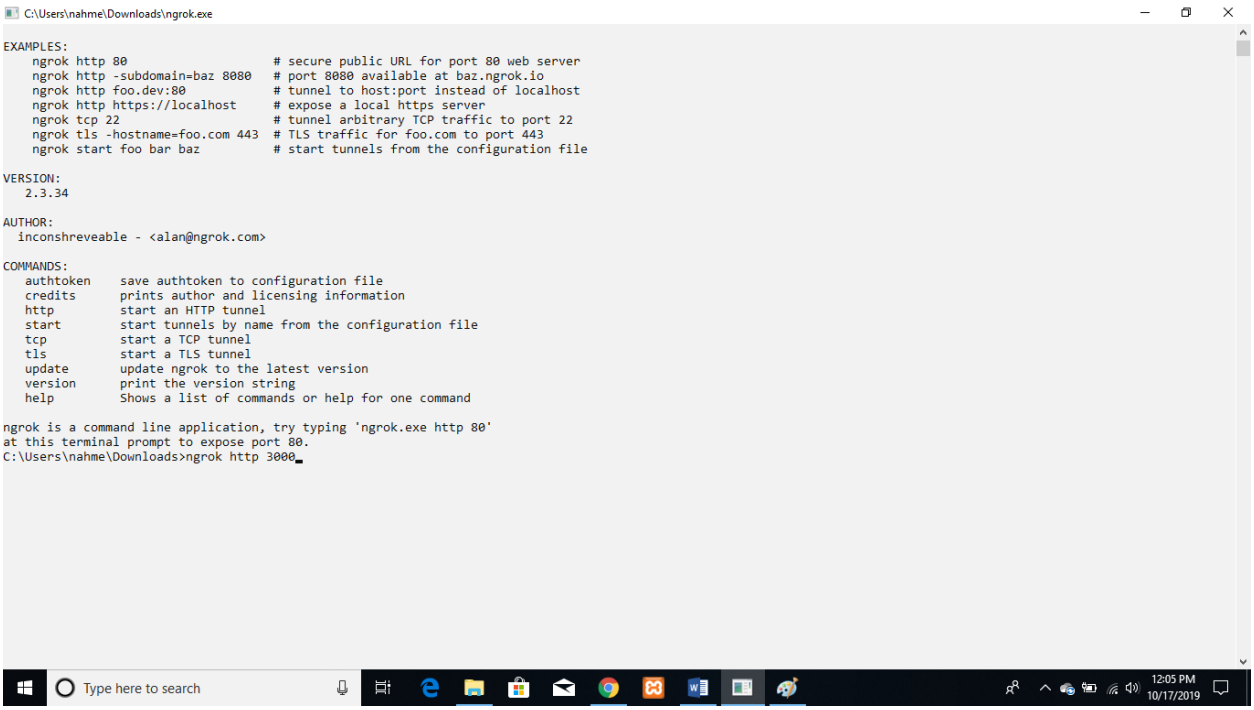
Task Number 3: Build a Shopify app with node and express

Follow this tutorial link provided by Shopify https://help.shopify.com/en/api/tutorials/build-a-shopify-app-with-node-and-express

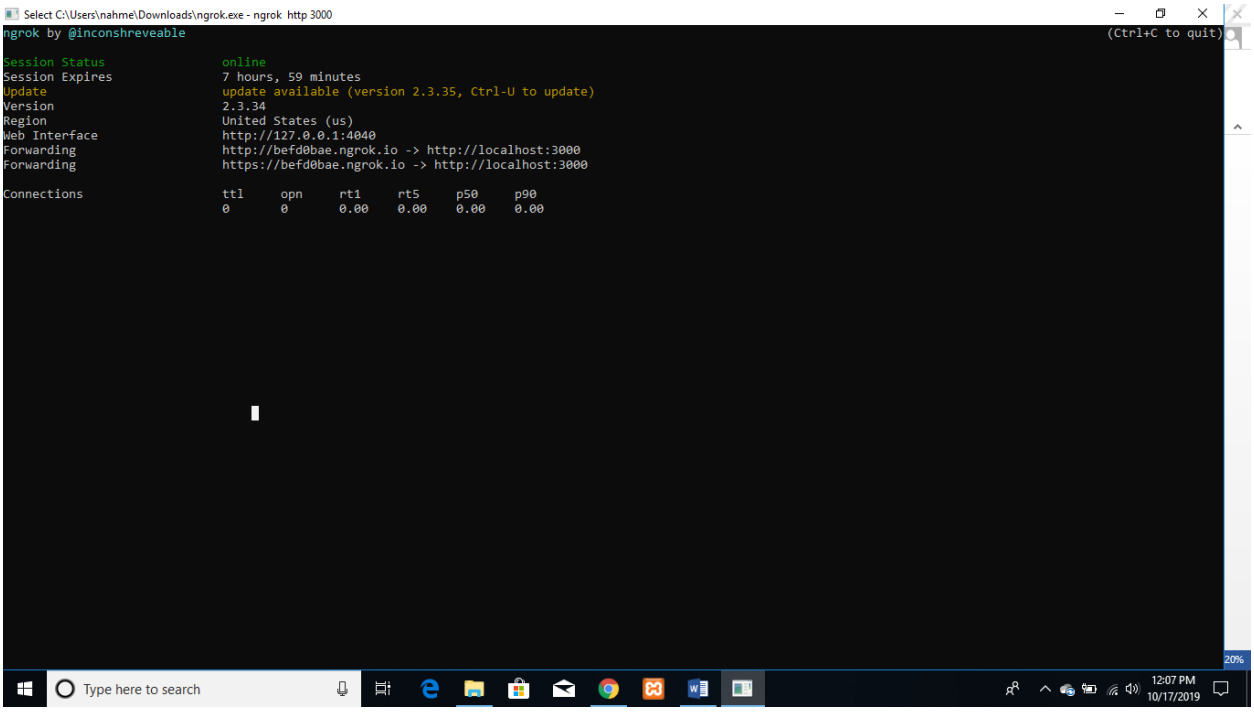
Step 1=> Expose your local development environment to the internet

Open ngrok.exe

Put=> ngrok http 3000



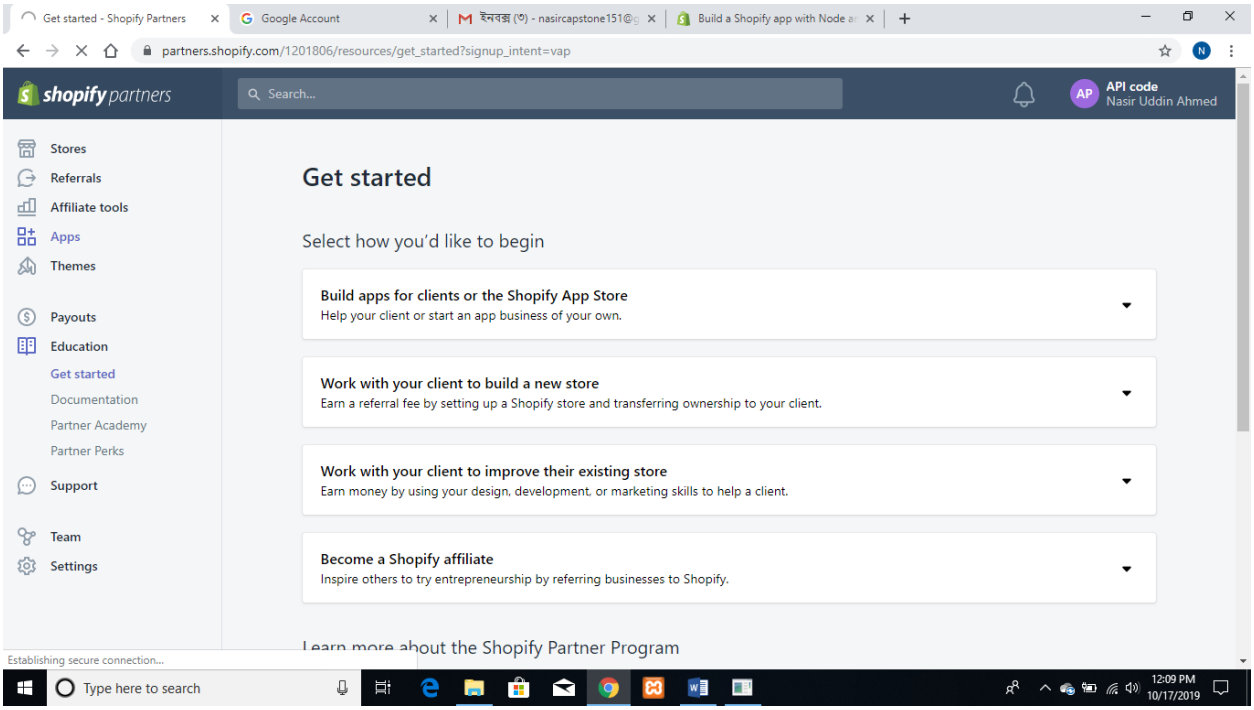
Press Enter



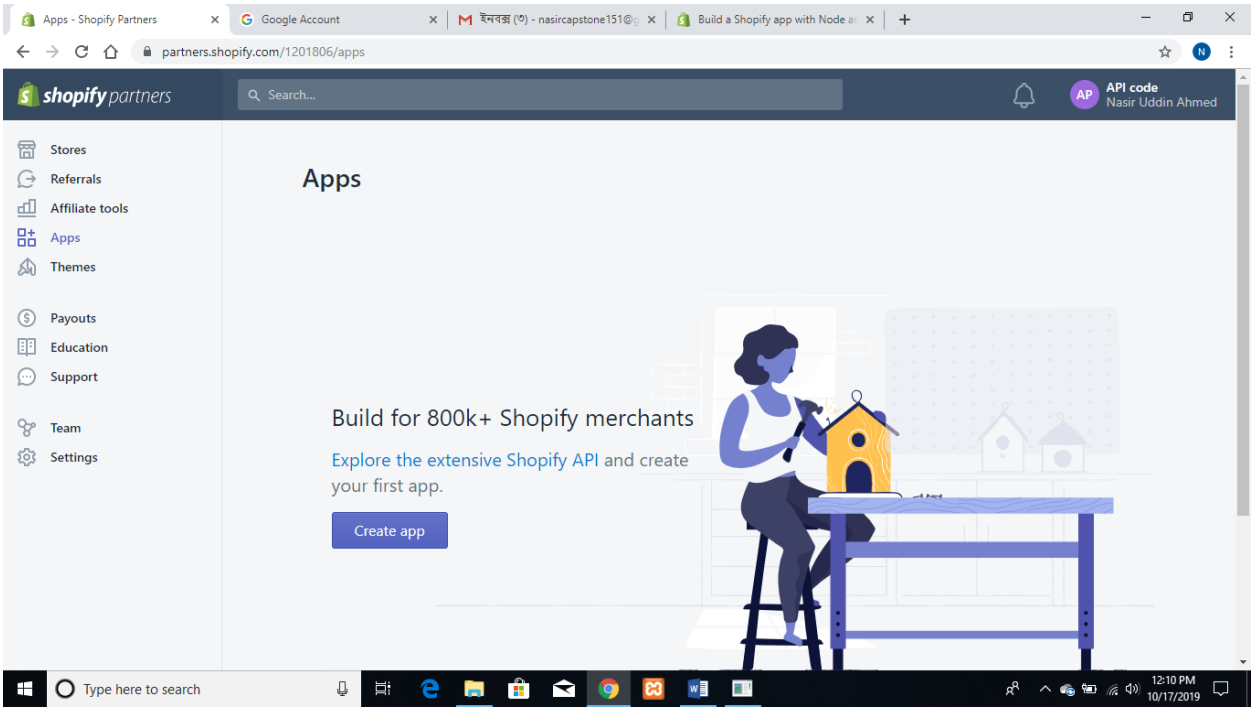
ngrok has generated tunneling address.

Step 2=> Create and configure your app in the Partner Dashboard

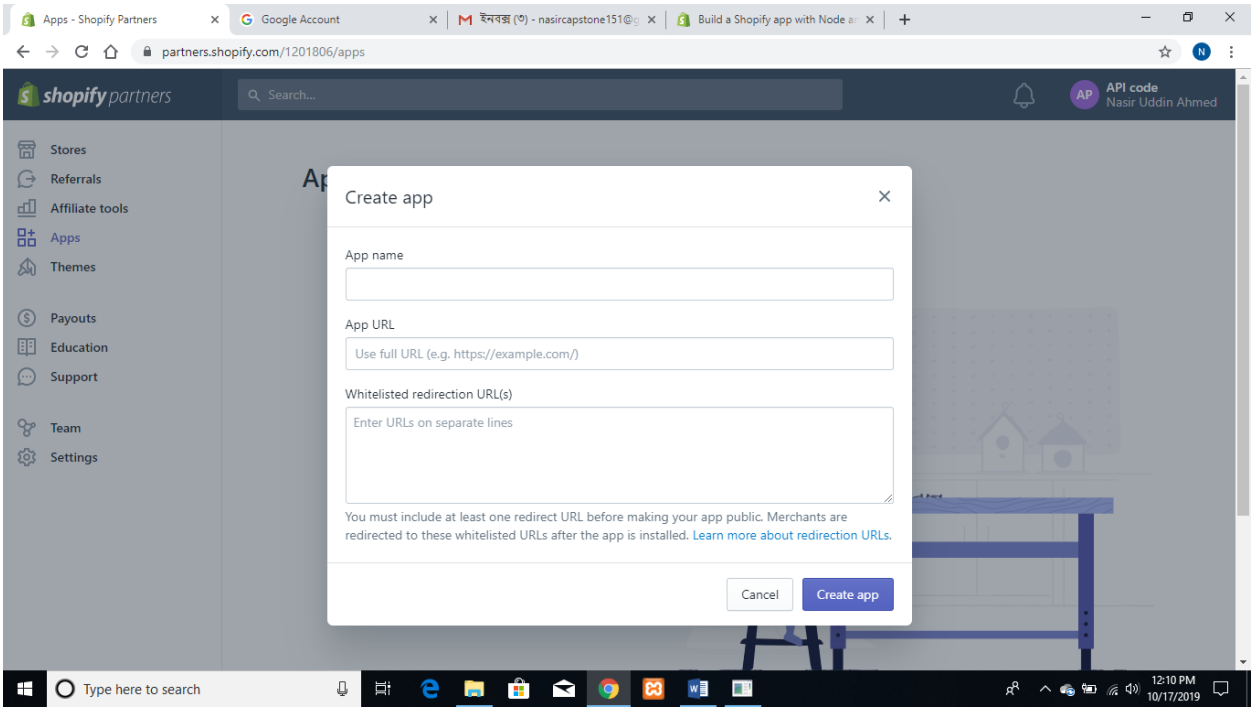
Go to your Shopify partner account. Press Apps



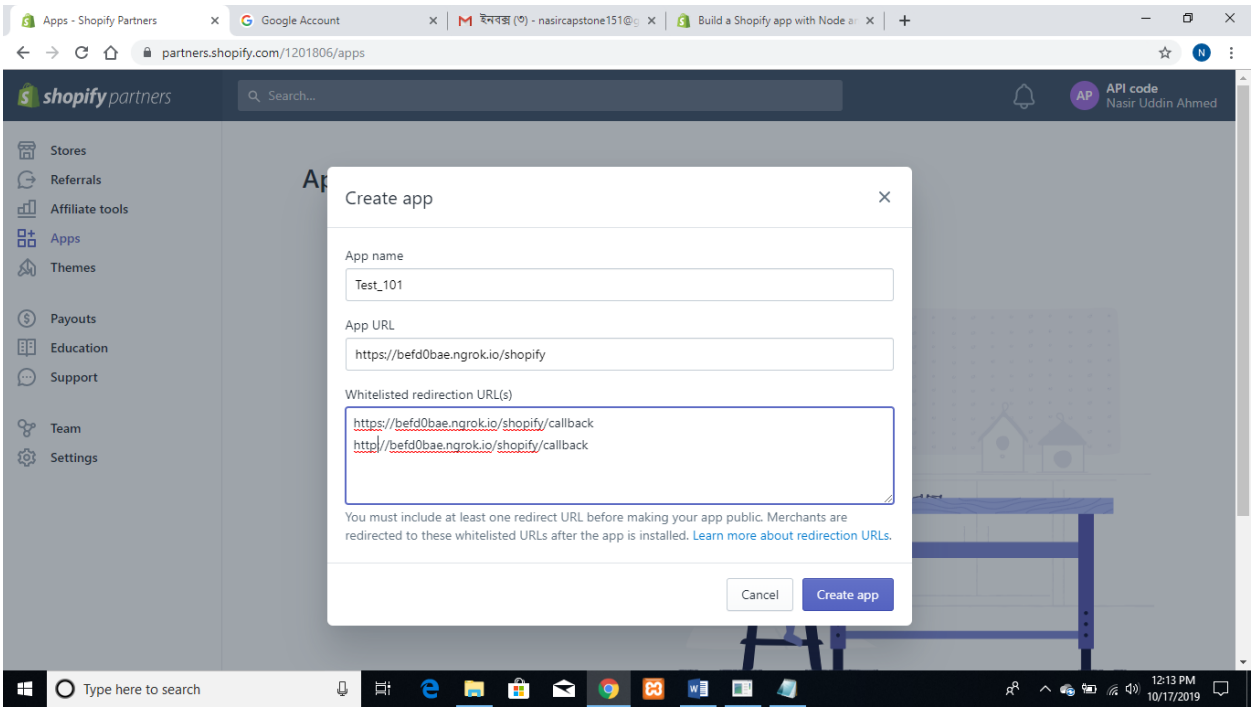
After pressing apps, a new window will appear.



Now press Create app



Here we need to put a App name, App URL generated by ngrok (Forwarding address). And Whitelisted redirection URL.

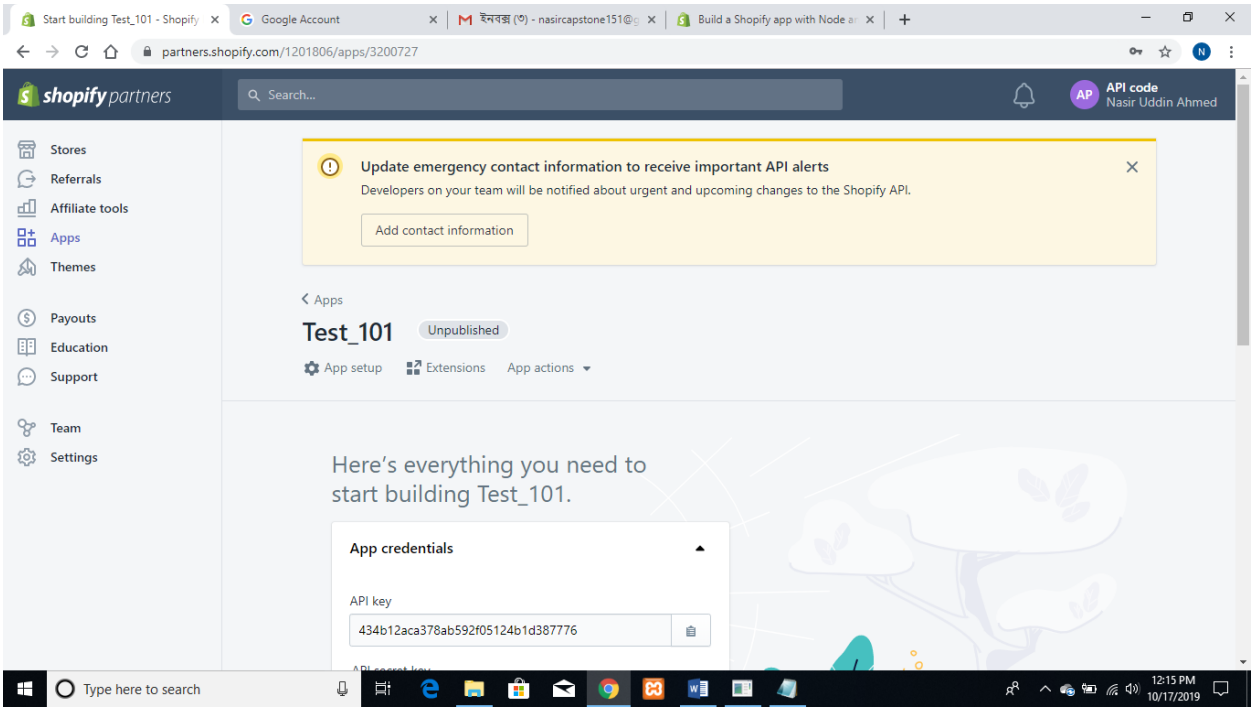


I have put my information according to my ngrok.

In App URL after .io put /shopify

In whitelisted URL after.io put /shopify/callback

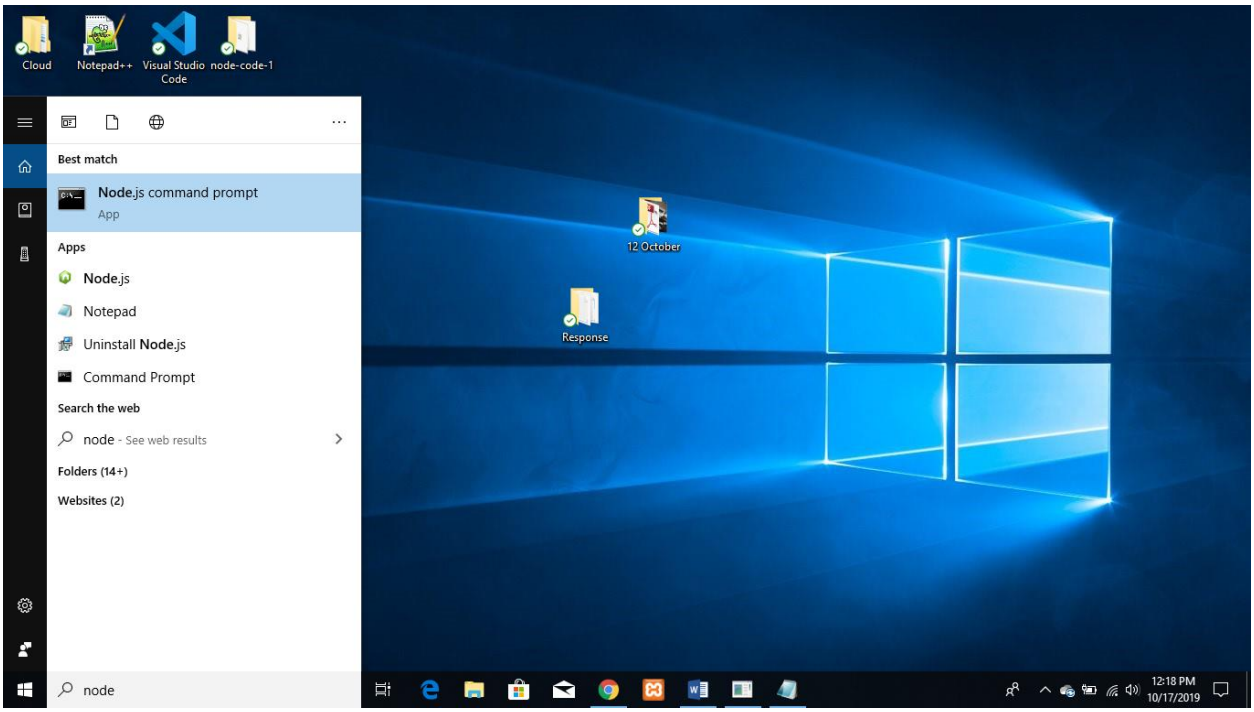
Now press Create app



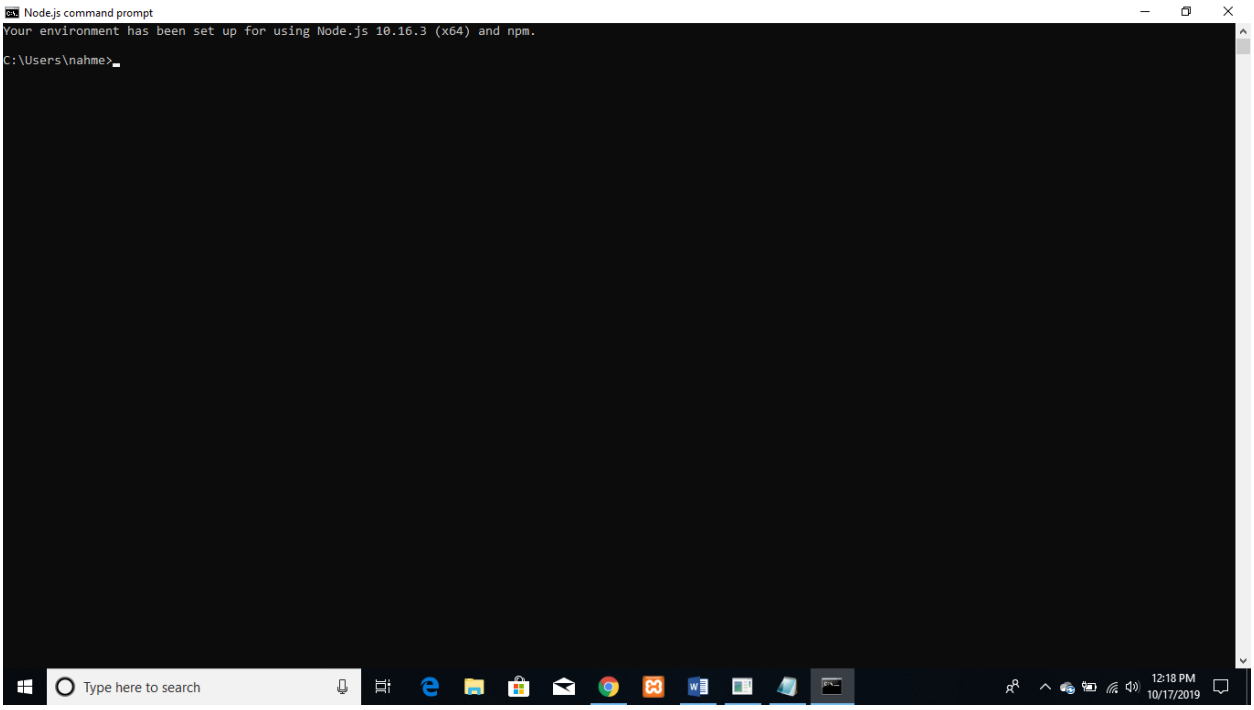
Here you will get your application's API Key, Secret key

Step 3=> Create a Node.js Project

Open Node.js command prompt



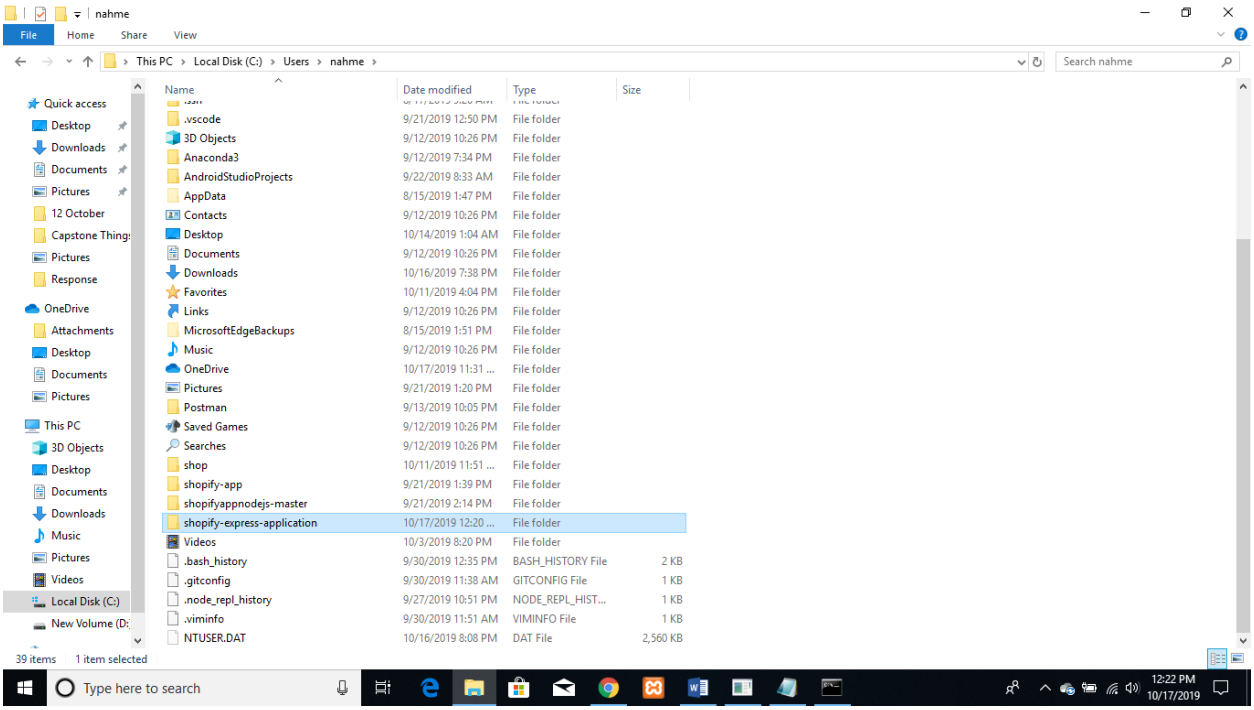
After clicking Node.js command prompt, the following window will come.



Now write 3 commands sequentially

Number 1: mkdir shopify-express-application

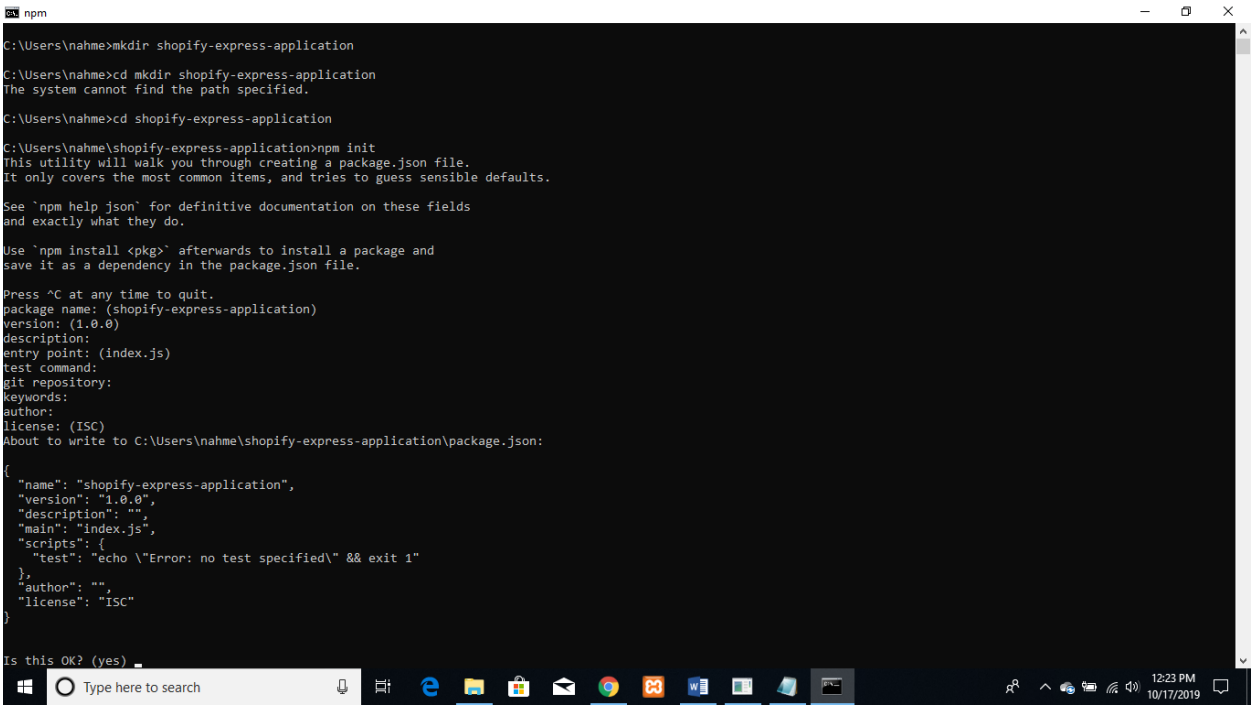
Number 2: cd shopify-express-application



Number 1 & Number 2 will create a Node.js project folder

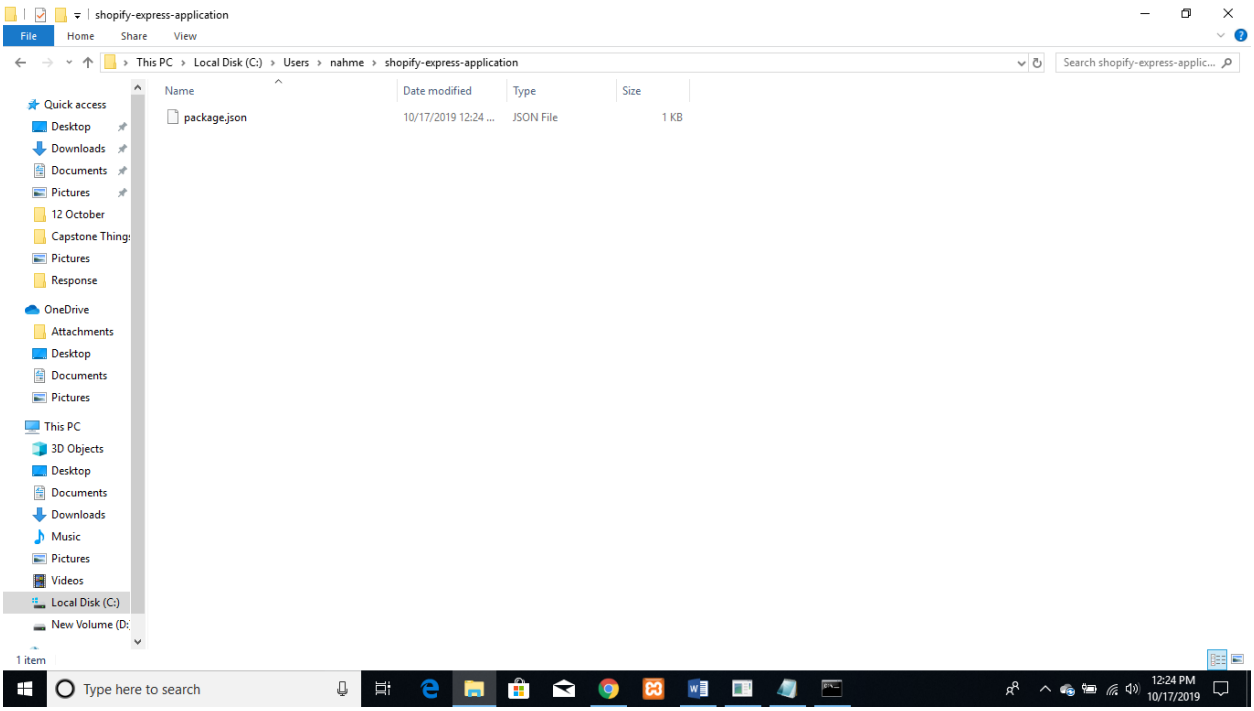
Number 3: npm init

After giving npm init ,



This will happen, input yes. And press enter.

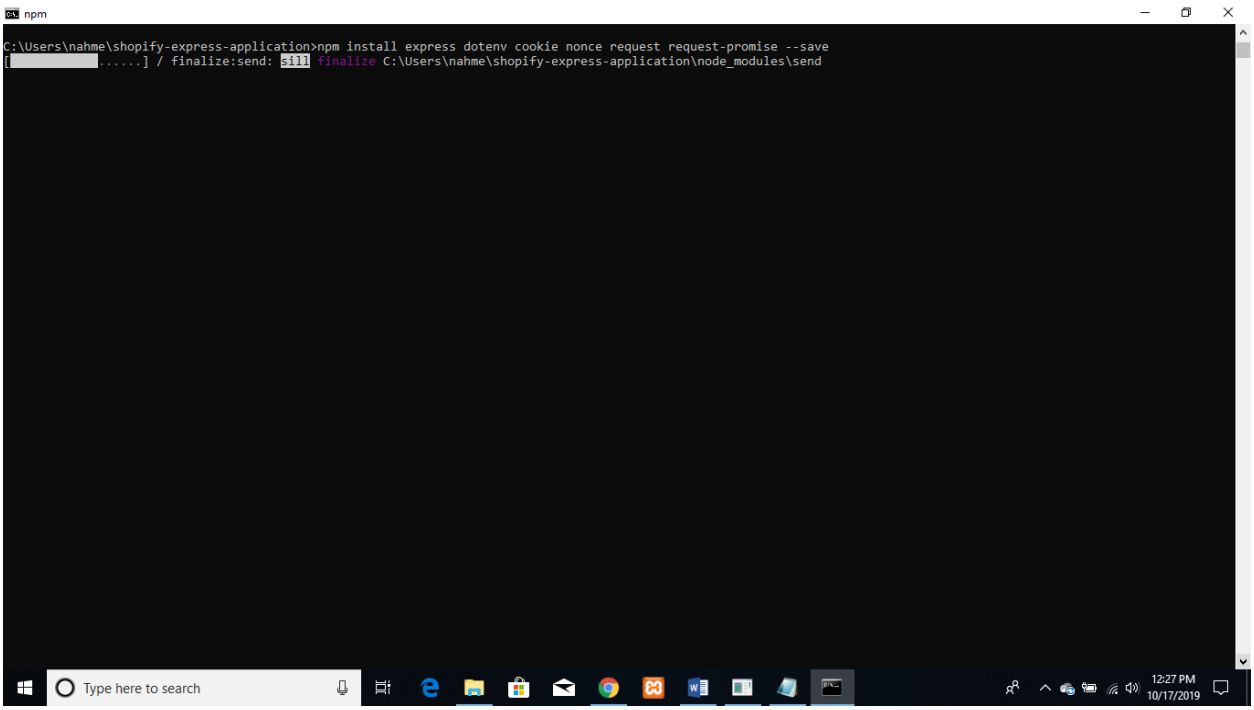
Then a package.json file would be generated inside the Node.js project folder.



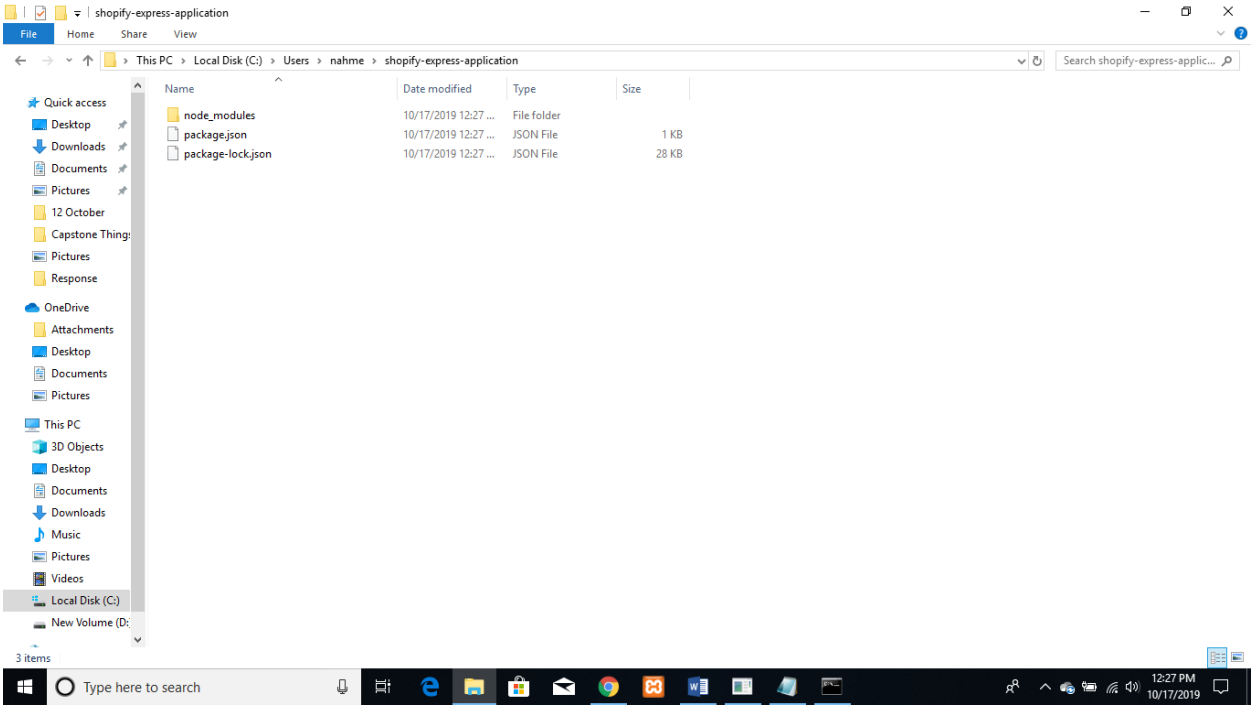
Documentation for npm init available in : https://docs.npmjs.com/cli/init

Now we need to install npm

Put npm install express dotenv cookie nonce request request-promise –save

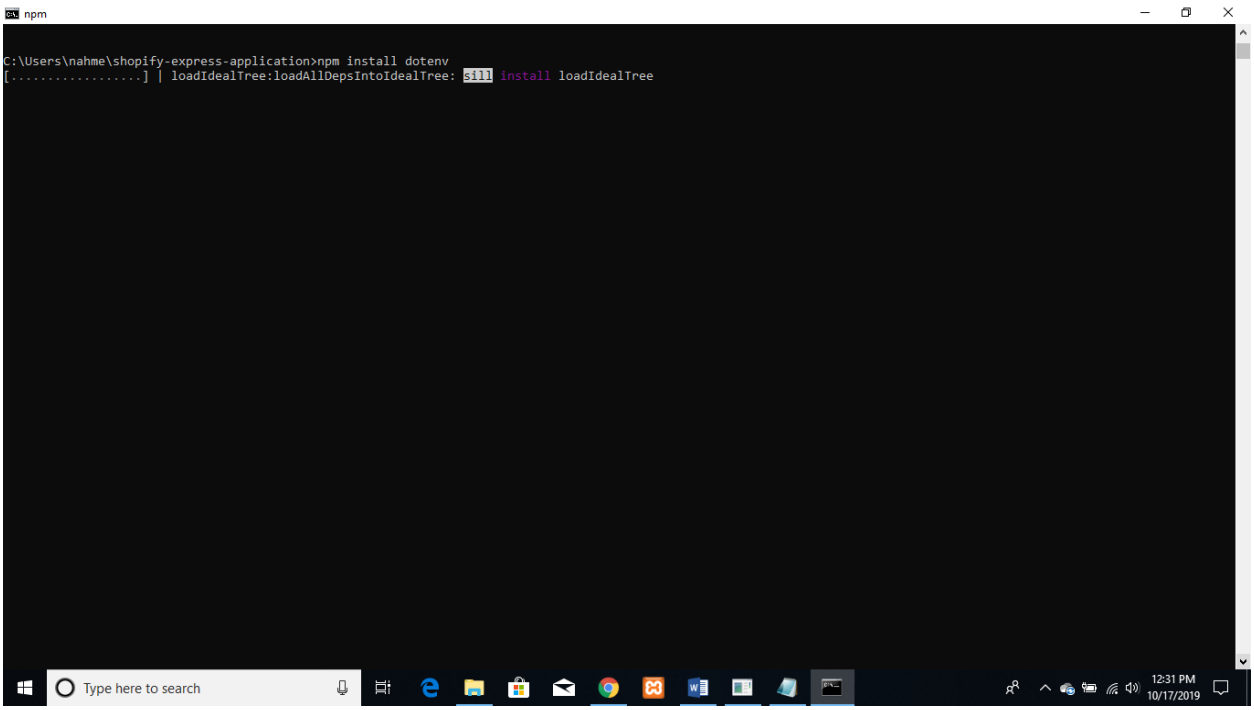


This command will install necessary node modules and package-lock.json,



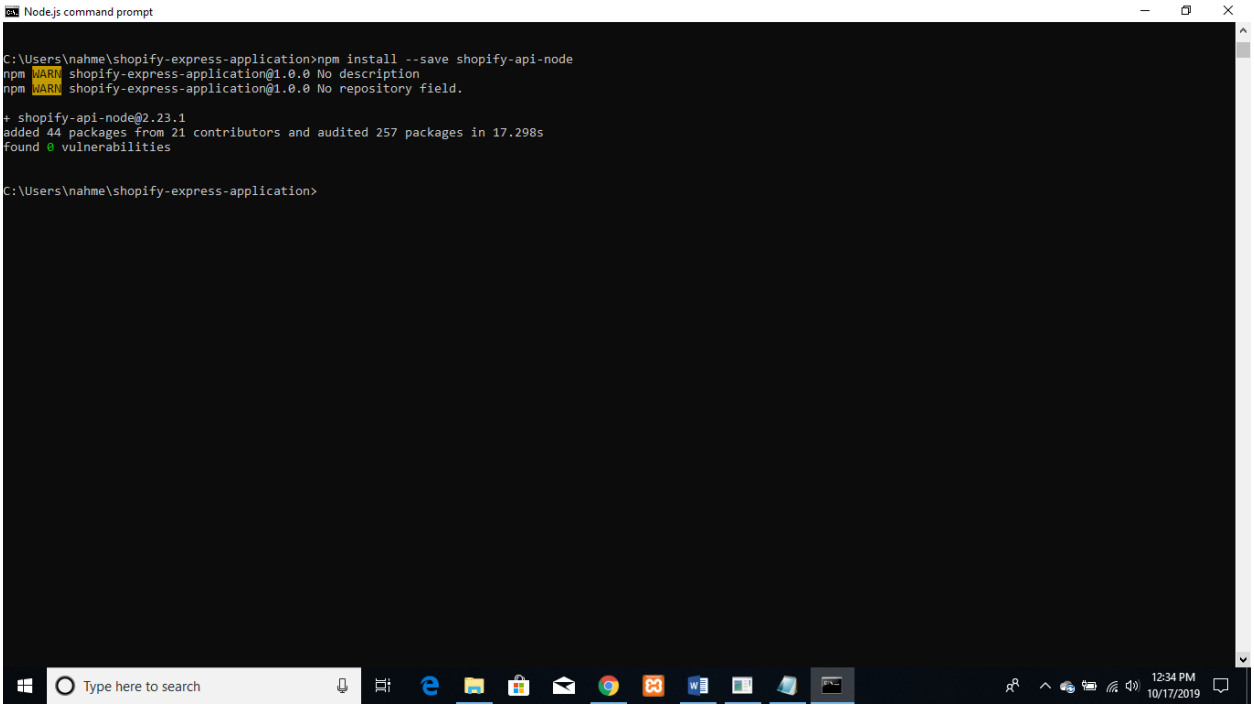
Put npm install dotenv

Documentation for dotenv is available : https://www.npmjs.com/package/dotenv



Put npm install --save shopify-api-node

Documentation for shopify-node-api available: https://www.npmjs.com/package/shopify-api-node



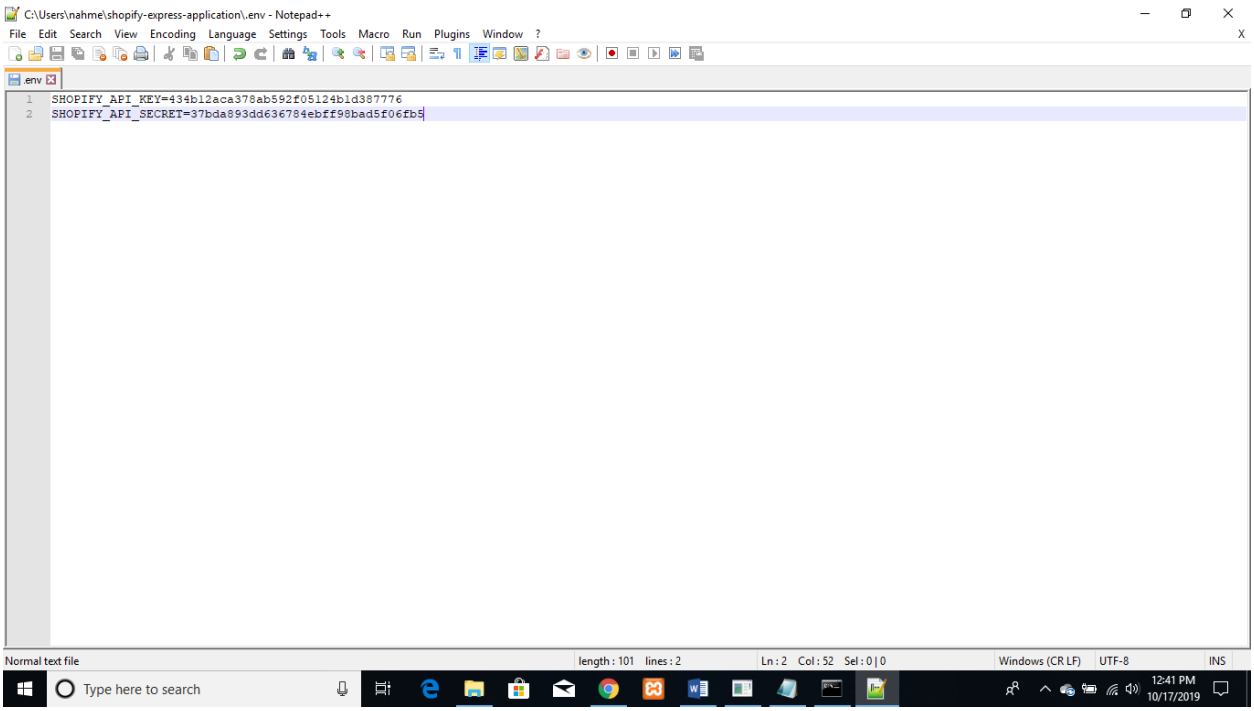
Now we need to create a .env file

Put touch .env (Sometimes touch would not work, it will show => touch is not recognized as an internal or external command, https://stackoverflow.com/questions/36126269/touch-is-not-recognized-as-an-internal-or-external-command-operable-program-o), in that case use npm install touch-cli –g

In the .env file, put

```
SHOPIFY_API_KEY="{YOUR_API_KEY}"
SHOPIFY_API_SECRET="{YOUR_API_SECRET_KEY}"
```

API KEY and API SECRET KEY available on Shopify partner account dashboard.



Now create a .gitignore file

Put touch .gitignore

In the .gitignore file type .env

Step 4=> Start building your Node.js app

Now create a index.js file

Put touch index.js

In the index.js file , copy the following code,

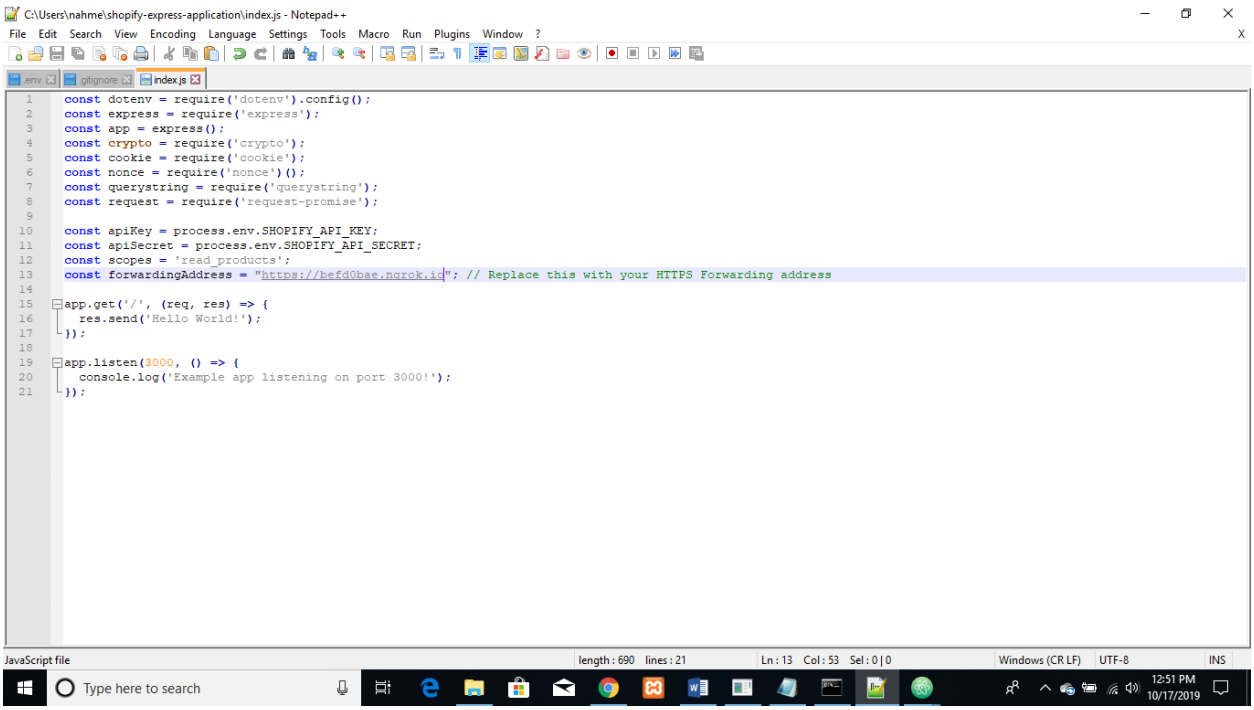
```
const dotenv = require('dotenv').config();
const express = require('express');
const app = express();
const crypto = require('crypto');
const cookie = require('cookie');
const nonce = require('nonce')();
const querystring = require('querystring');
const request = require('request-promise');
```

```
const apiKey = process.env.SHOPIFY_API_KEY;
const apiSecret = process.env.SHOPIFY_API_SECRET;
const scopes = 'read_products';
const forwardingAddress = "{ngrok forwarding address}"; // Replace this with
your HTTPS Forwarding address

app.get('/', (req, res) => {
  res.send('Hello World!');
});

app.listen(3000, () => {
  console.log('Example app listening on port 3000!');
});
```

Here=>

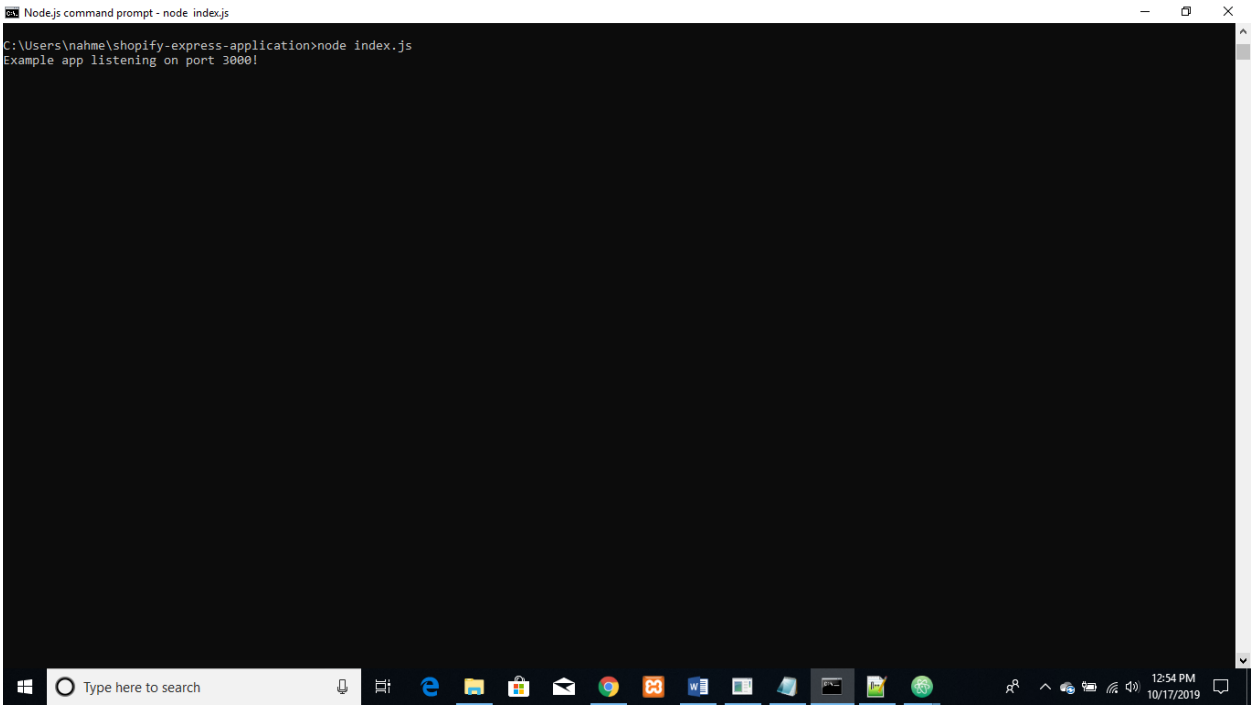
`const forwardingAddress = "{ngrok forwarding address}";` replace the forwarding address, put the address that has been generated by ngrok.

Your index.js code file will look like



Now you need to run the Node.js application to see the tunneling.

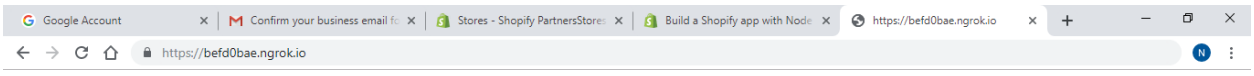
Put node index.js



Node index.js would run the application.

Go to your browser and type the forwarding address generated by ngrok, (https://befd0bae.ngrok.io)

It will show a Hello World String.



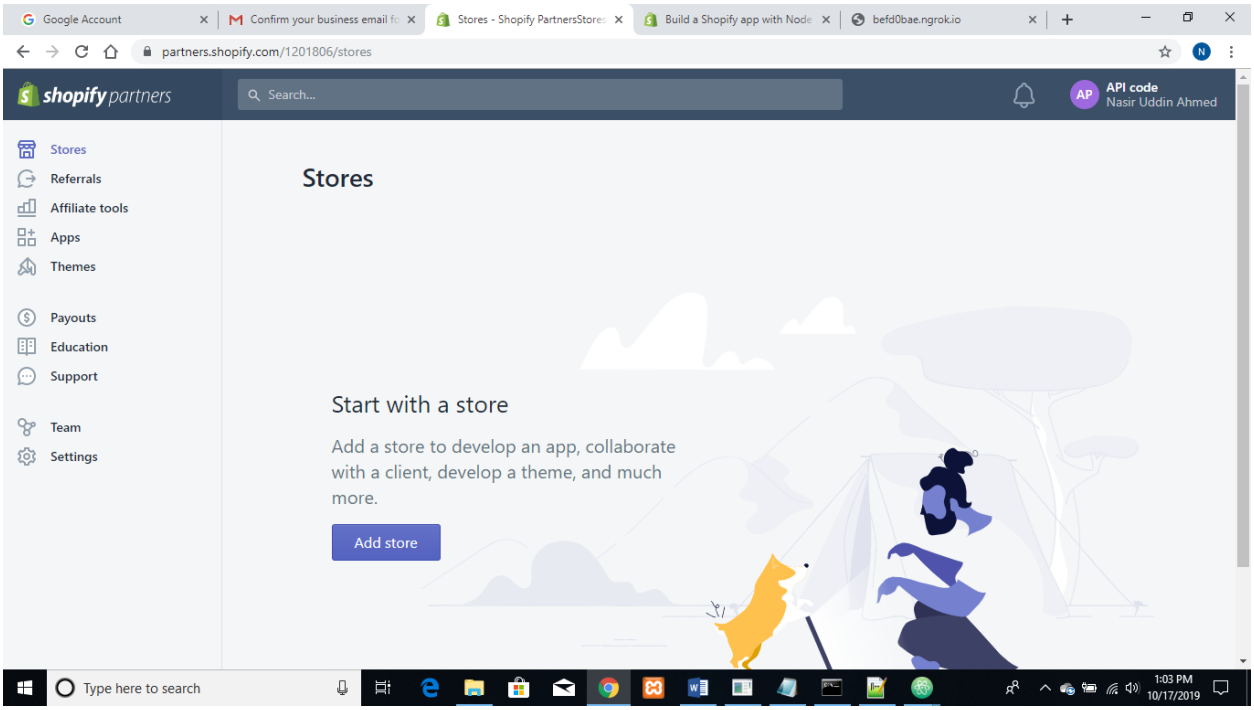
Hello World!

If the code fails, then

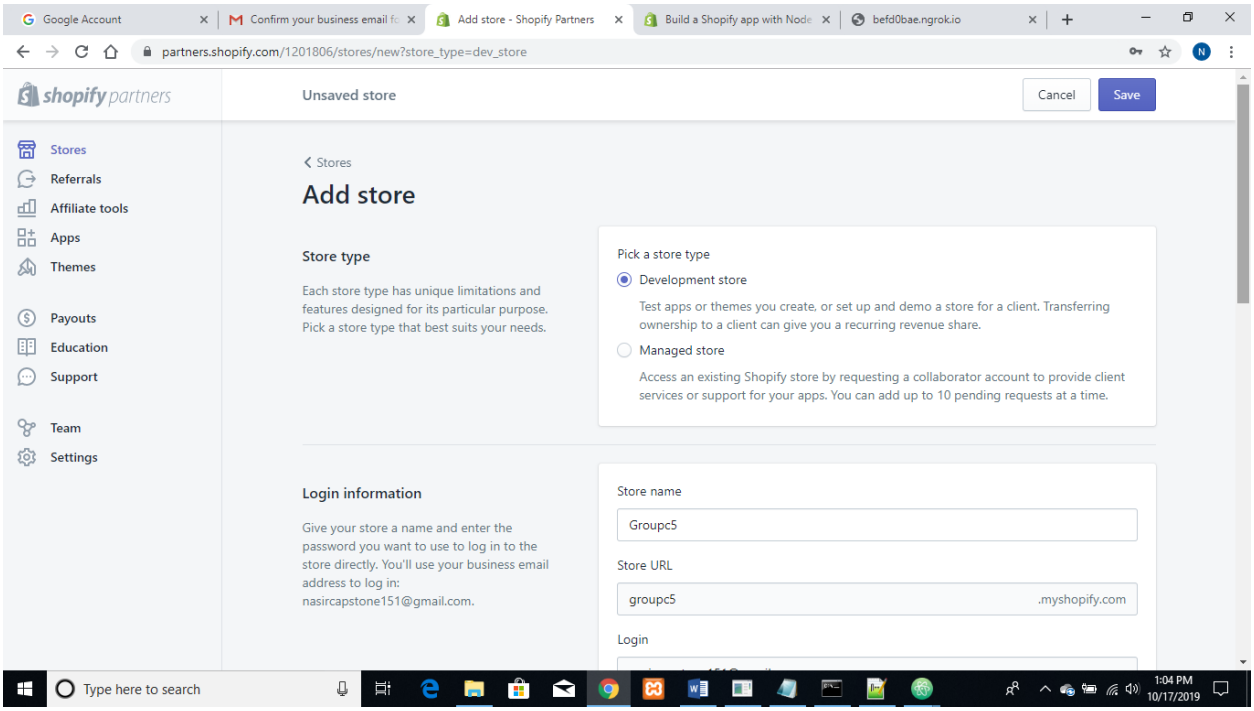
```
const scopes = 'read_products';  replace this line with const scopes = 'write_products';
in the index.js file, then save and run the projet by using node index.js command from Node.js command prompt
```

Step 5=> Create Shopify-specific routes

Before creating specific route we need to set a Shopify store. Go to Shopify partner dashboard, click on Add store.

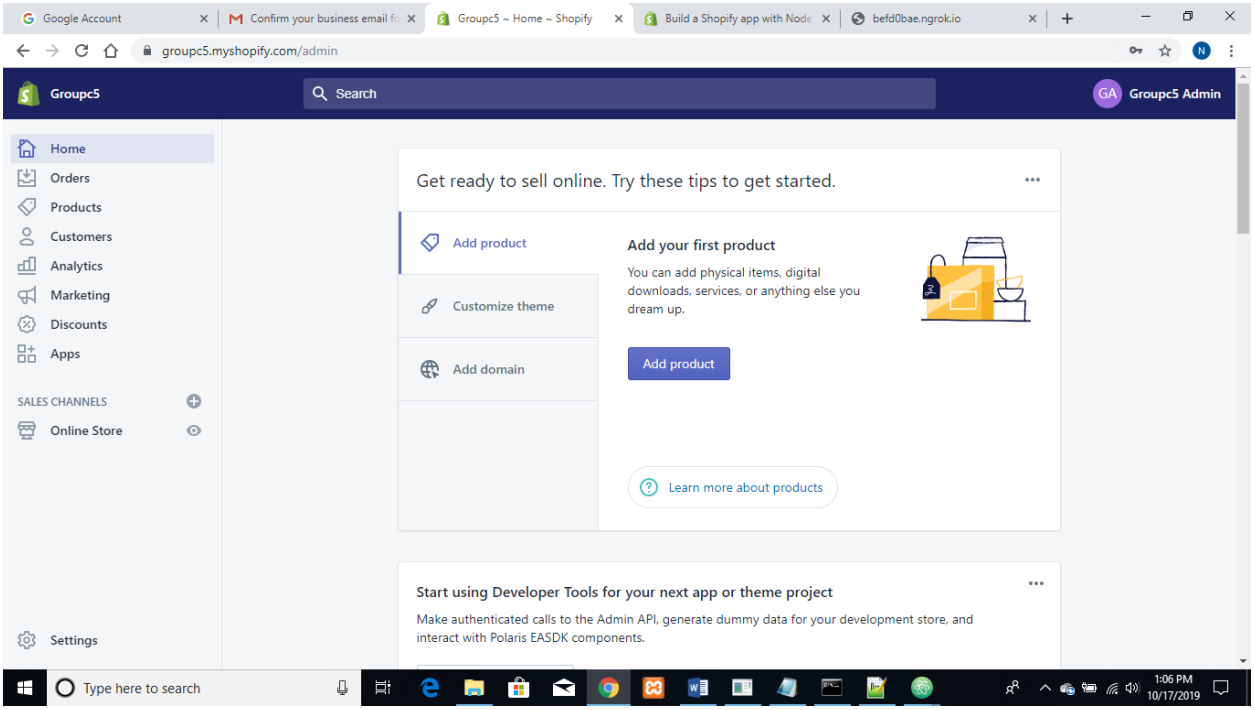


Fill up necessary information, shown below. Here I gave a store name, you can give yours.



Then press save

You will be redirected to your store's admin panel.



Now go to index.js file, replace the previous code with the given code below

```
const dotenv = require('dotenv').config();
const express = require('express');
const app = express();
const crypto = require('crypto');
const cookie = require('cookie');
const nonce = require('nonce')();
const querystring = require('querystring');
const request = require('request-promise');

const apiKey = process.env.SHOPIFY_API_KEY;
const apiSecret = process.env.SHOPIFY_API_SECRET;
const scopes = 'write_products'; // both privileg
const forwardingAddress = "https://5dec032c.ngrok.io"; // Replace this with your HTTPS Forwarding address

/*app.get('/', (req, res) => {
  res.send('Hello World!, I Am Nasir Uddin Ahmed From Capstone C5 Group, I Am A Software Engineer');
});*/

app.get('/shopify', (req, res) => {
  const shop = req.query.shop;
  if (shop) {
    const state = nonce(); // constant noce
    const redirectUri = forwardingAddress + '/shopify/callback';
    const installUrl = 'https://' + shop +
      '/admin/oauth/authorize?client_id=' + apiKey +
      '&scope=' + scopes +
      '&state=' + state +
      '&redirect_uri=' + redirectUri; // hit back , source
```

```
    res.cookie('state', state);
    res.redirect(installUrl);
  } else {
    return res.status(400).send('Missing shop parameter. Please add ?shop=your-development-shop.myshopify.com to your request');
  }
});



/*app.get('/shopify/callback', (req, res) => {
  const { shop, hmac, code, state } = req.query;
  const stateCookie = cookie.parse(req.headers.cookie).state;

  if (state !== stateCookie) {
    return res.status(403).send('Request origin cannot be verified');
  }

  if (shop && hmac && code) {
    res.status(200).send('Callback route');

    // TODO
    // Validate request is from Shopify
    // Exchange temporary code for a permanent access token
      // Use access token to make API call to 'shop' endpoint
      // Custom coding started form here
      const map=Object.assign({}, req.query); // Custom Coding
      delete map['hmac']; // Custom Coding
      const message=querystring.stringify(map); // Custom Coding
```

```
    const generatedHash=crypto
    .createHmac('sha256',apiSecret) // my secret apiKey
    .update(message)
    .digest('hex');

    if(generatedHash!==hmac){
      return res.status(400).send('HMAC Validaion failed');
    }
    return res.status(200).send('HMAC Validated');



  } else {
    res.status(400).send('Required parameters missing');
  }
});*/
app.get('/shopify/callback', (req, res) => {
  const { shop, hmac, code, state } = req.query;
  const stateCookie = cookie.parse(req.headers.cookie).state;

  if (state !== stateCookie) {
    return res.status(403).send('Request origin cannot be verified');
  }

  if (shop && hmac && code) {
    // DONE: Validate request is from Shopify
    const map = Object.assign({}, req.query);
    delete map['signature'];
    delete map['hmac'];
```

```
const message = querystring.stringify(map);
const providedHmac = Buffer.from(hmac, 'utf-8');
const generatedHash = Buffer.from(
  crypto
    .createHmac('sha256', apiSecret)
    .update(message)
    .digest('hex'),
    'utf-8'
  );
let hashEquals = false;

try {
  hashEquals = crypto.timingSafeEqual(generatedHash, providedHmac)
} catch (e) {
  hashEquals = false;
};

if (!hashEquals) {
  return res.status(400).send('HMAC validation failed');
}

// DONE: Exchange temporary code for a permanent access token
const accessTokenRequestUrl = 'https://' + shop + '/admin/oauth/access_token';
const accessTokenPayload = {
  client_id: apiKey,
  client_secret: apiSecret,
  code,
};
```

```
    request.post(accessTokenRequestUrl, { json: accessTokenPayload })
    .then((accessTokenResponse) => {
      const accessToken = accessTokenResponse.access_token;
      // DONE: Use access token to make API call to 'shop' endpoint
      const shopRequestUrl = 'https://' + shop + '/admin/api/2019-04/products.json';
      //const shopRequestUrl = 'https://' + shop + '/admin/shop.json';
      const shopRequestHeaders = {
        'X-Shopify-Access-Token': accessToken, // Verified
      };

      request.get(shopRequestUrl, { headers: shopRequestHeaders })
      .then((shopResponse) => {
        res.status(200).end(shopResponse);
      })
      .catch((error) => {
        res.status(error.statusCode).send(error.error.error_description);
      });
    })
    .catch((error) => {
      res.status(error.statusCode).send(error.error.error_description);
    });

  } else {
    res.status(400).send('Required parameters missing');
  }
});


app.listen(3000, () => {
```

```
  console.log('Example app listening on port 3000!');
});
```

Careful , don't forget to change the forwarding address.

```
const forwardingAddress = "https://5dec032c.ngrok.io"; // Replace this with your HTTPS Forwarding address
```

Save index.js

And now run the index.js file again by using node index.js command.

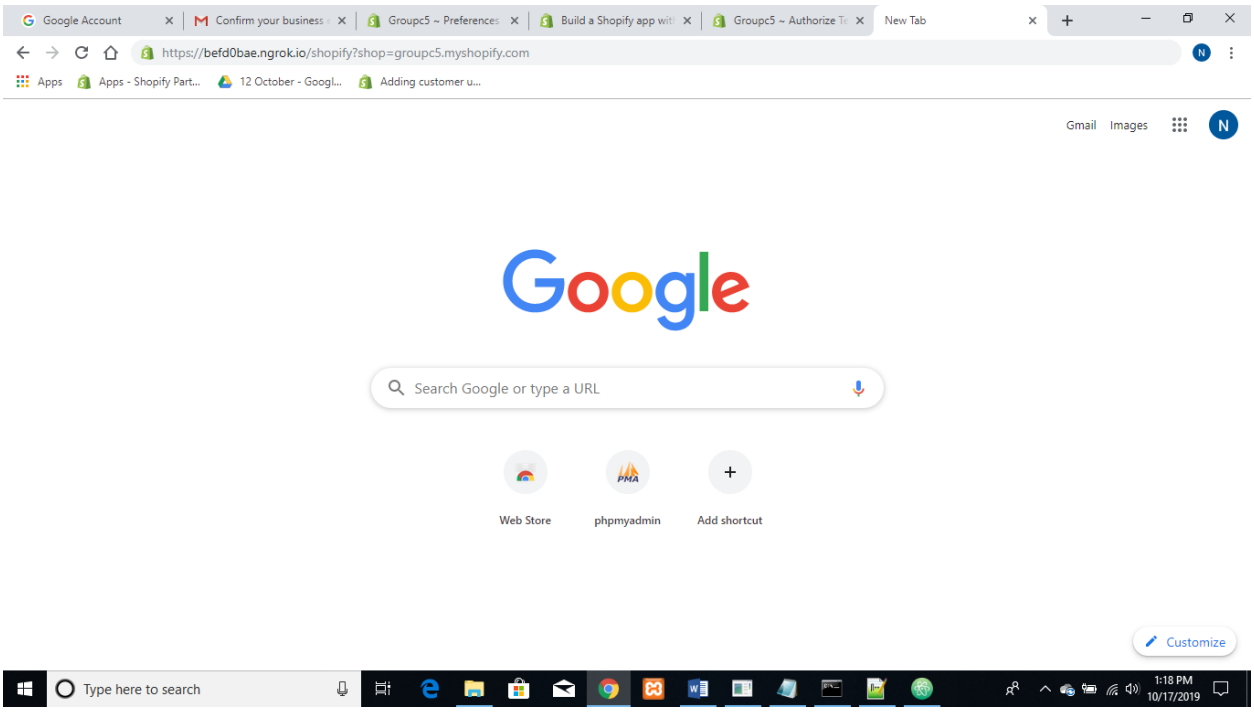
Step 6=> Run the application

Go to your browser

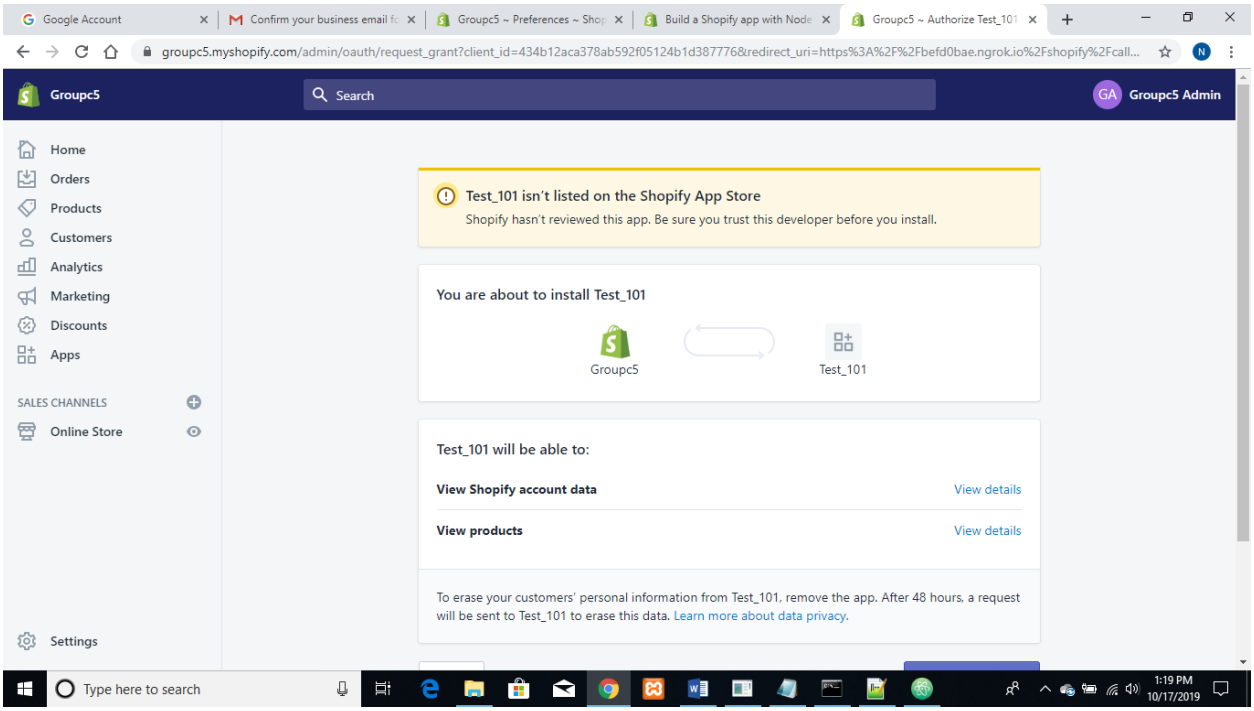
`{your ngrok forwarding address}/shopify?shop=your-development-shop.myshopify.com`

For our project it looked like

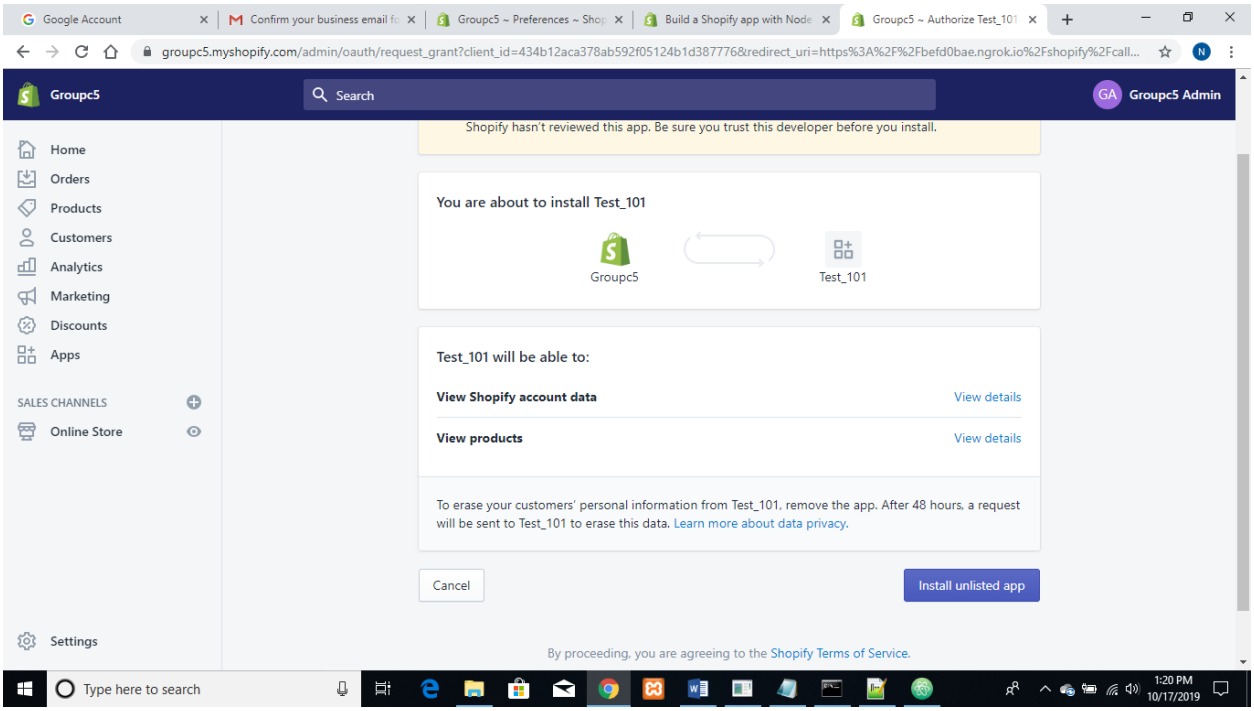
https://befd0bae.ngrok.io/shopify?shop=groupc5.myshopify.com



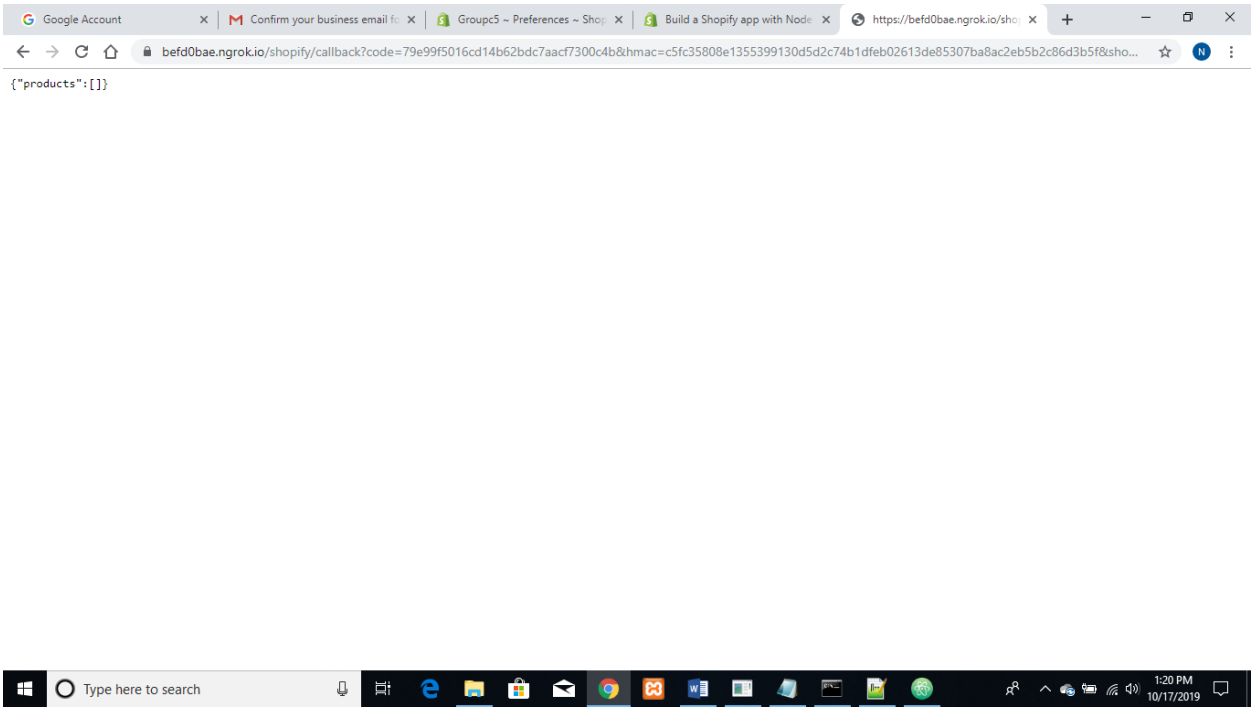
Put that link that has been showed in the above picture. Wait for some time. Tunneling forwarding needs time to detect.



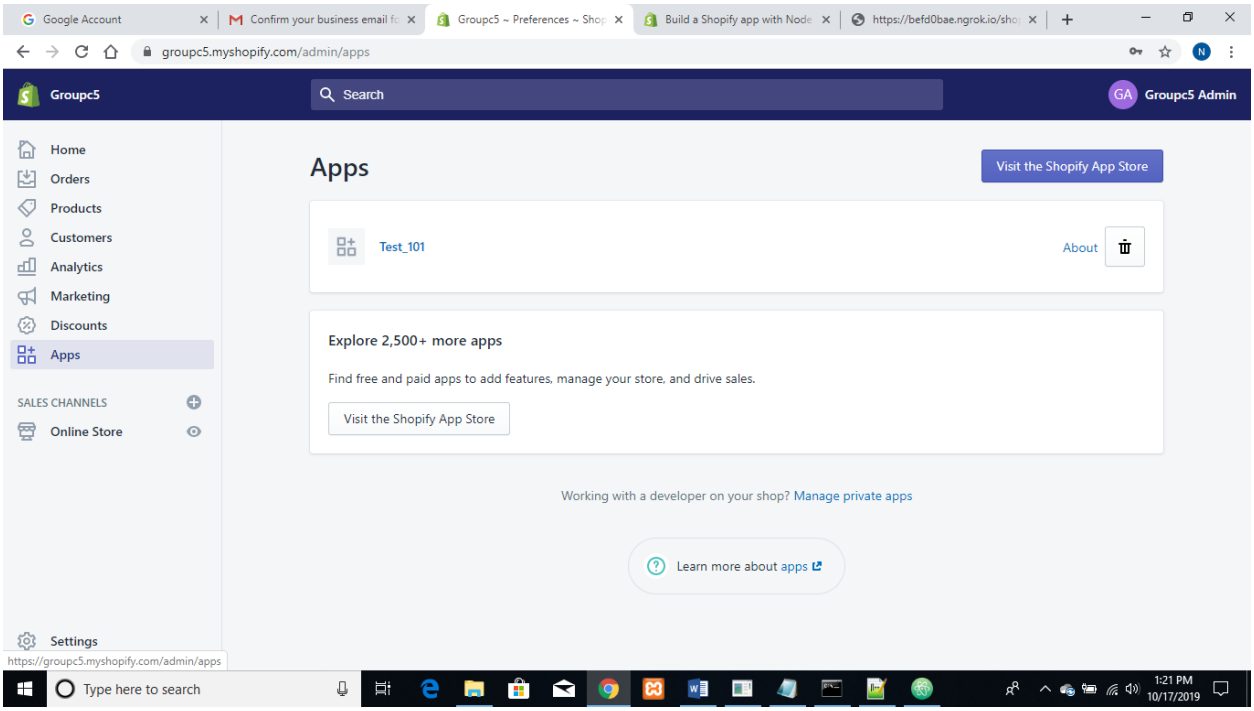
If all ok, congratulations!!!, you have successfully created a bridge, forwarding for you Shopify application.



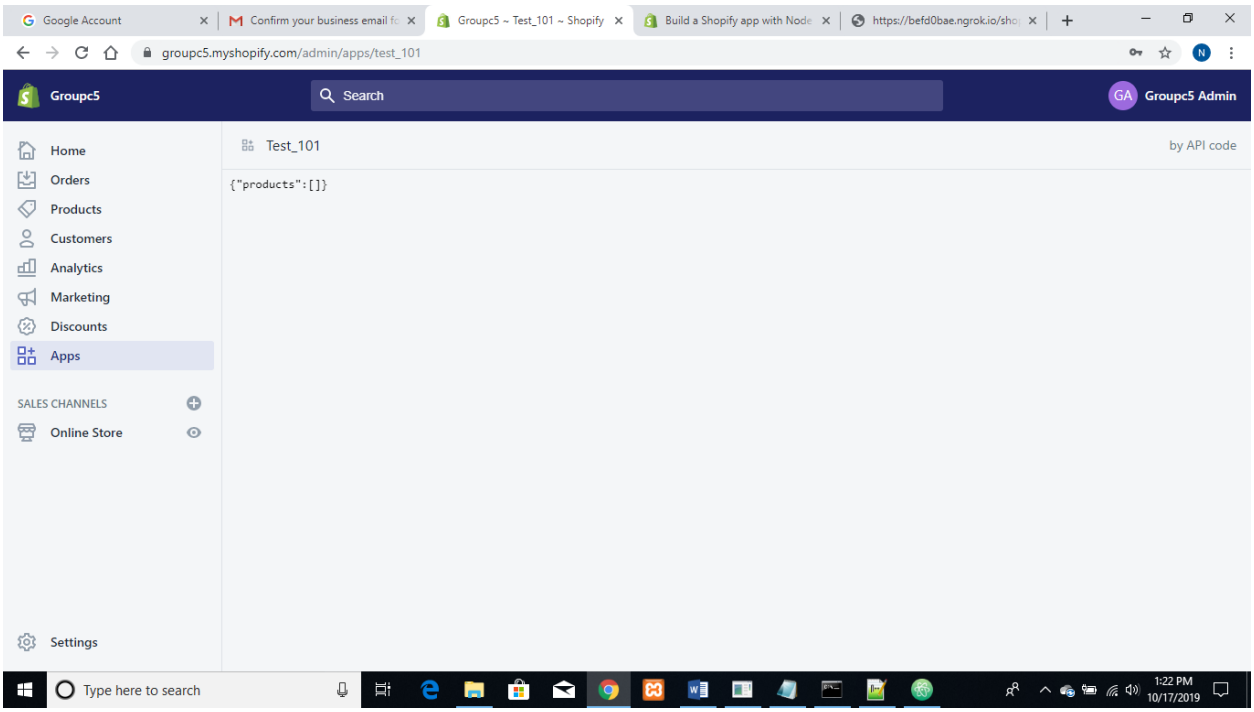
Press Install unlisted app



The page will show empty product list, as there is no product in the store.

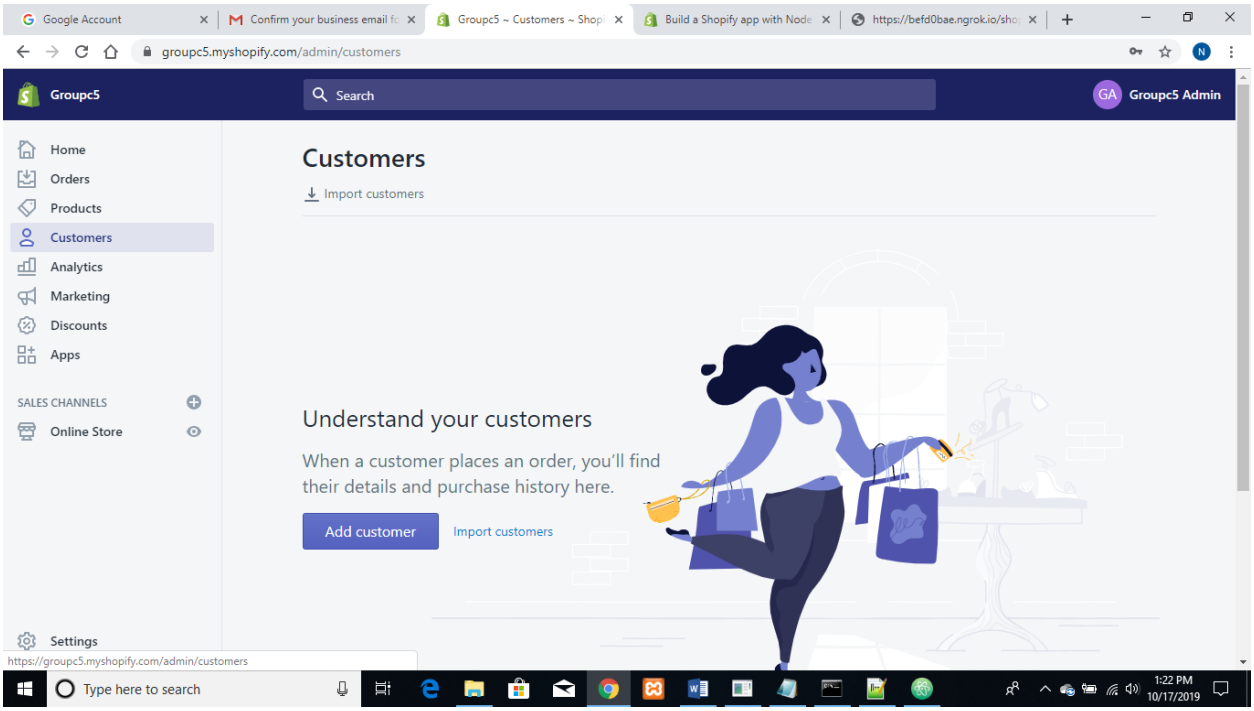


By clicking Apps, you can see the application



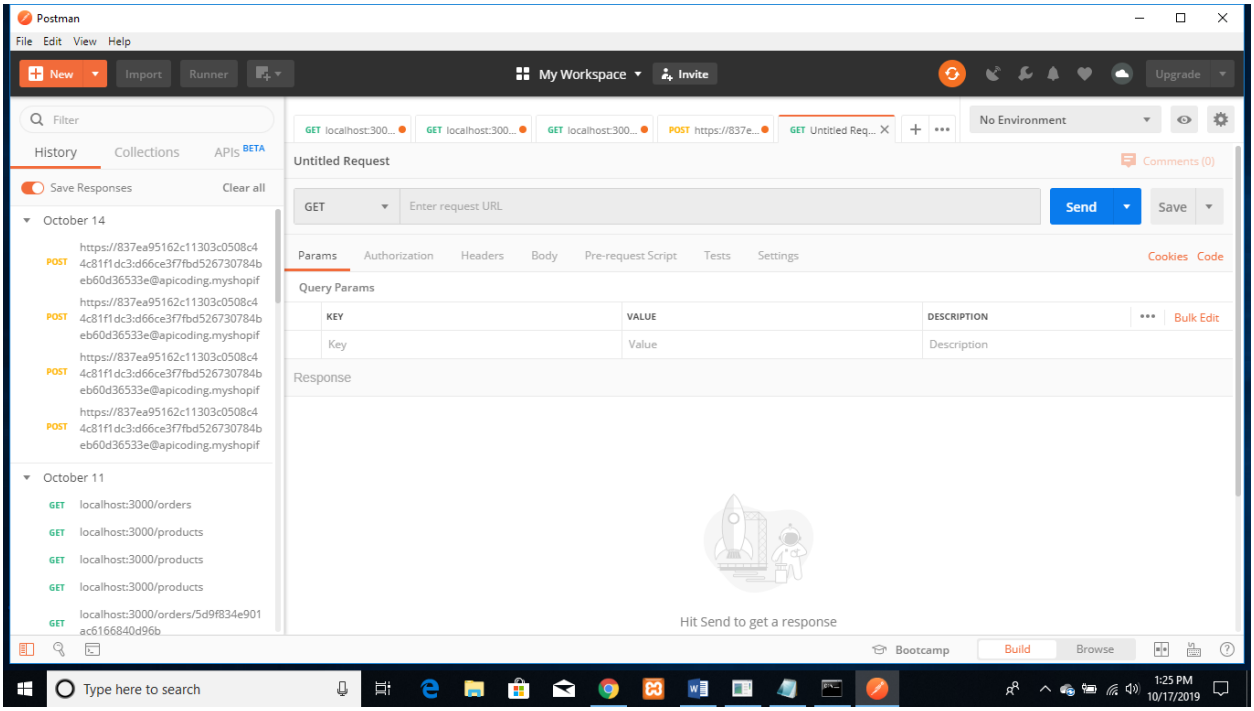
Now test the app, using User registration.

User Registration By API Calling



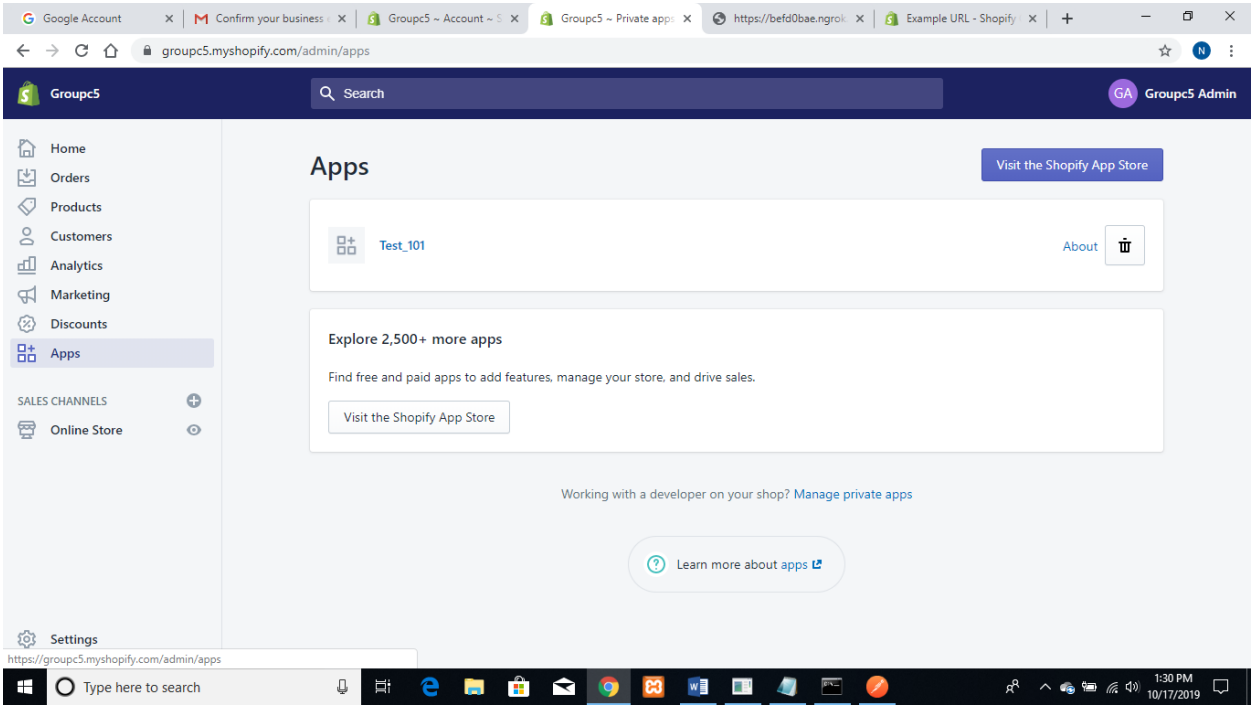
By clicking customers, we can see there is no customer at present. To call the API, we need postman.

Open postman app.

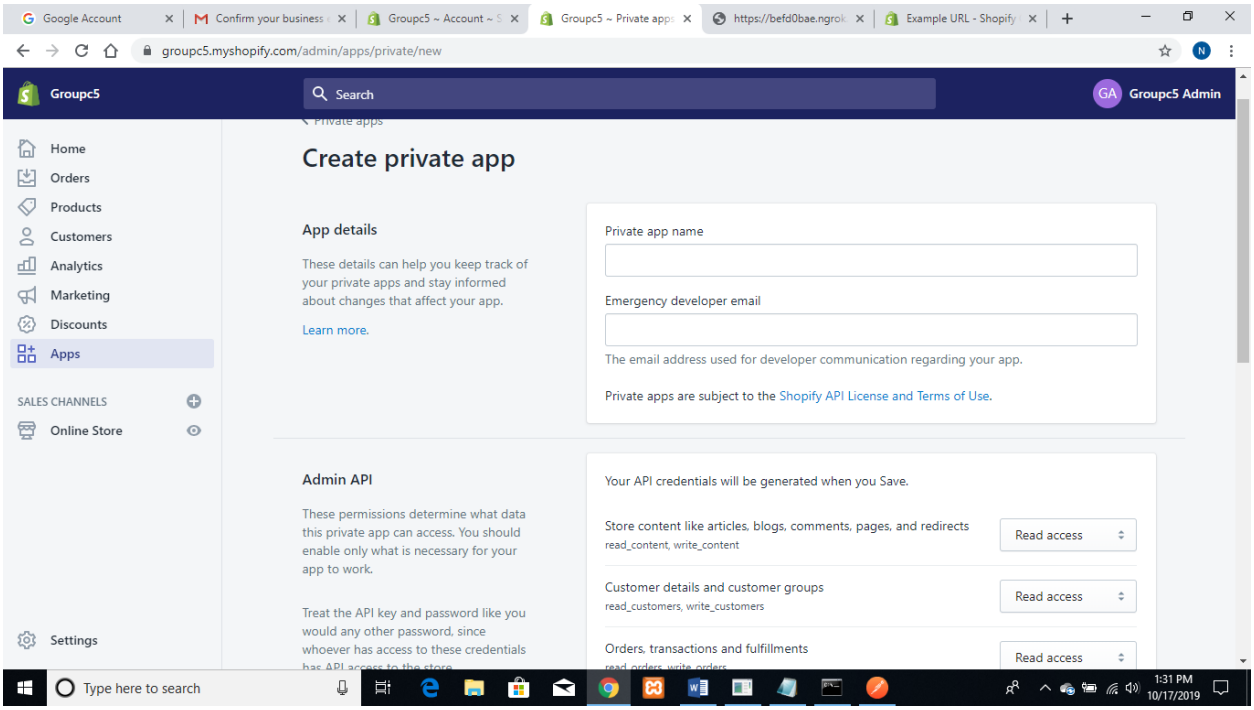


In the postman app, we have to select POST request, (because we want to register a user)

Go to https://accounts.shopify.com/store-login, here you will get example URL, that is consisted with access token, and credentials.



Click on manage private apps.

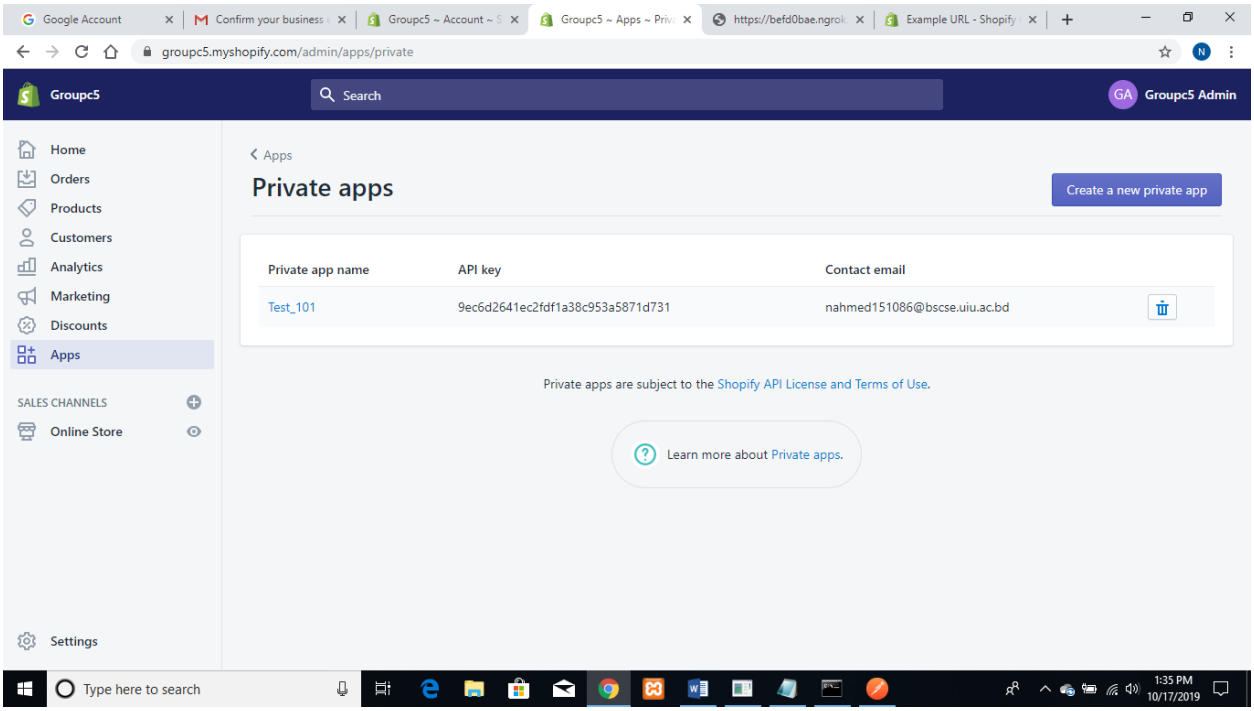


Give the App name that you used before, In our case it was Test_101

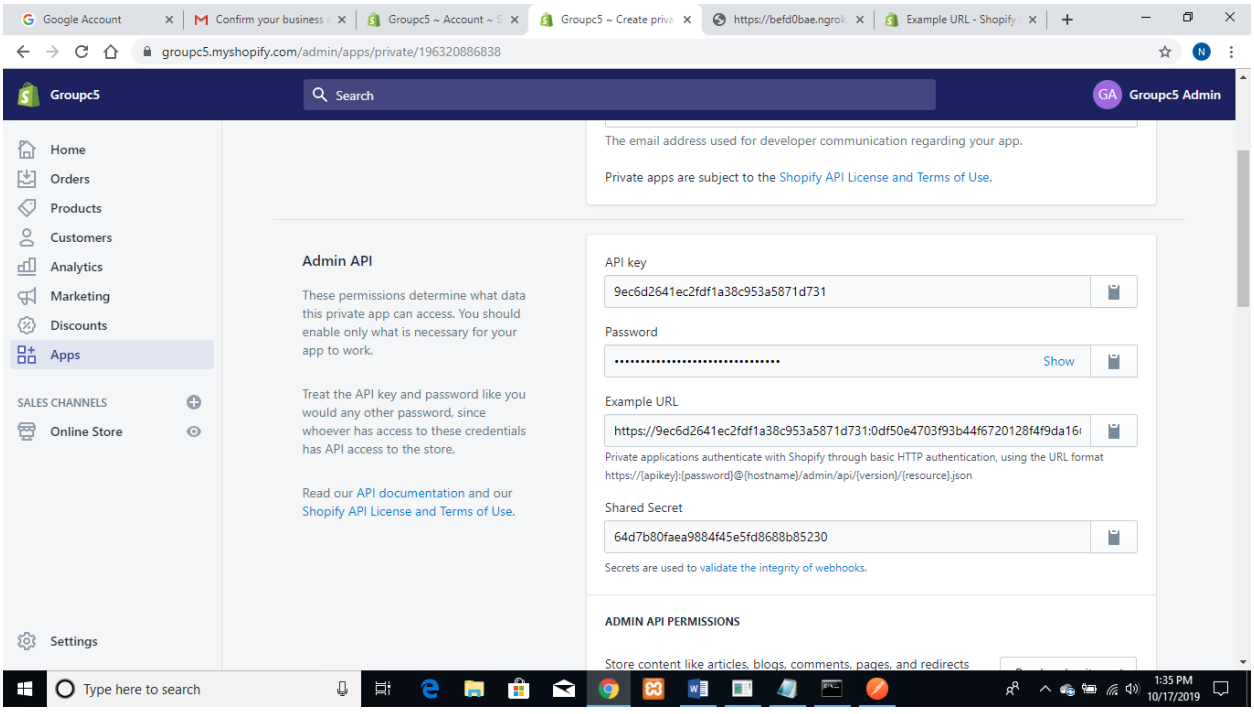
Make the read access to Read and Write Access.

Put a tik mark on Allow this app to access your storefront data using the Storefront API. Press save.

Click the Apps option, there you will see your app. Click app(Test_101).

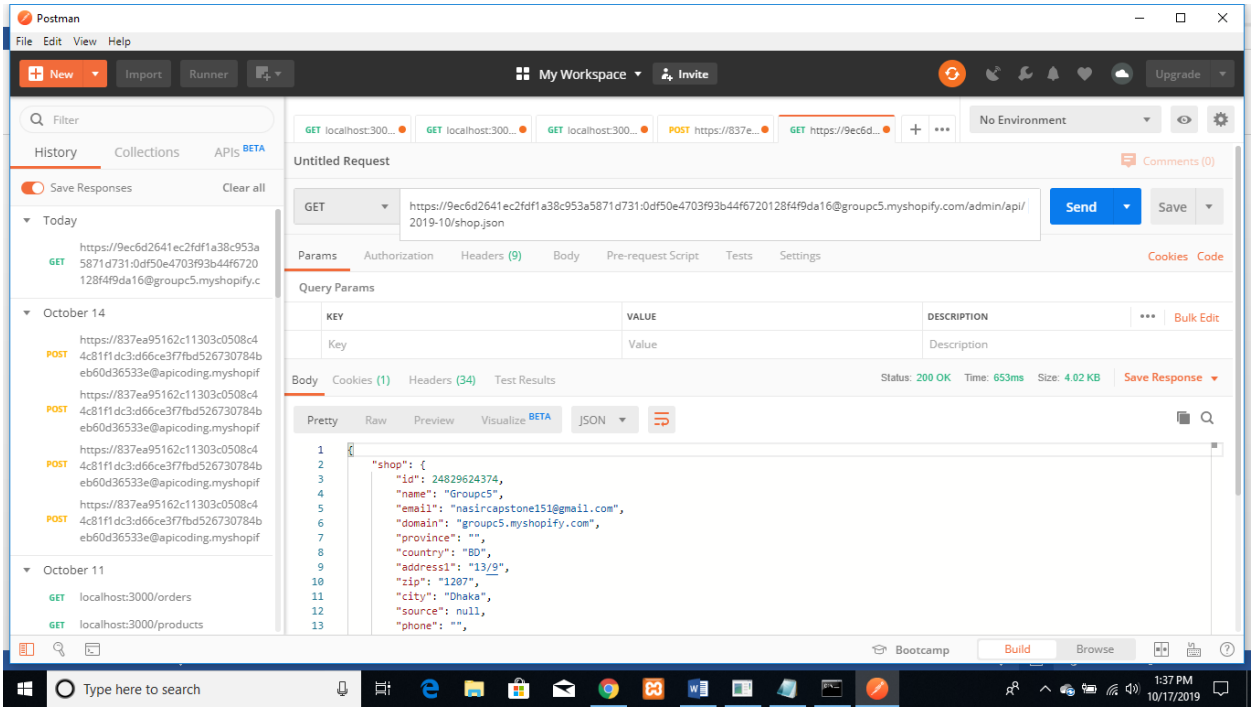


Clicking on Test_101 will show you a new page



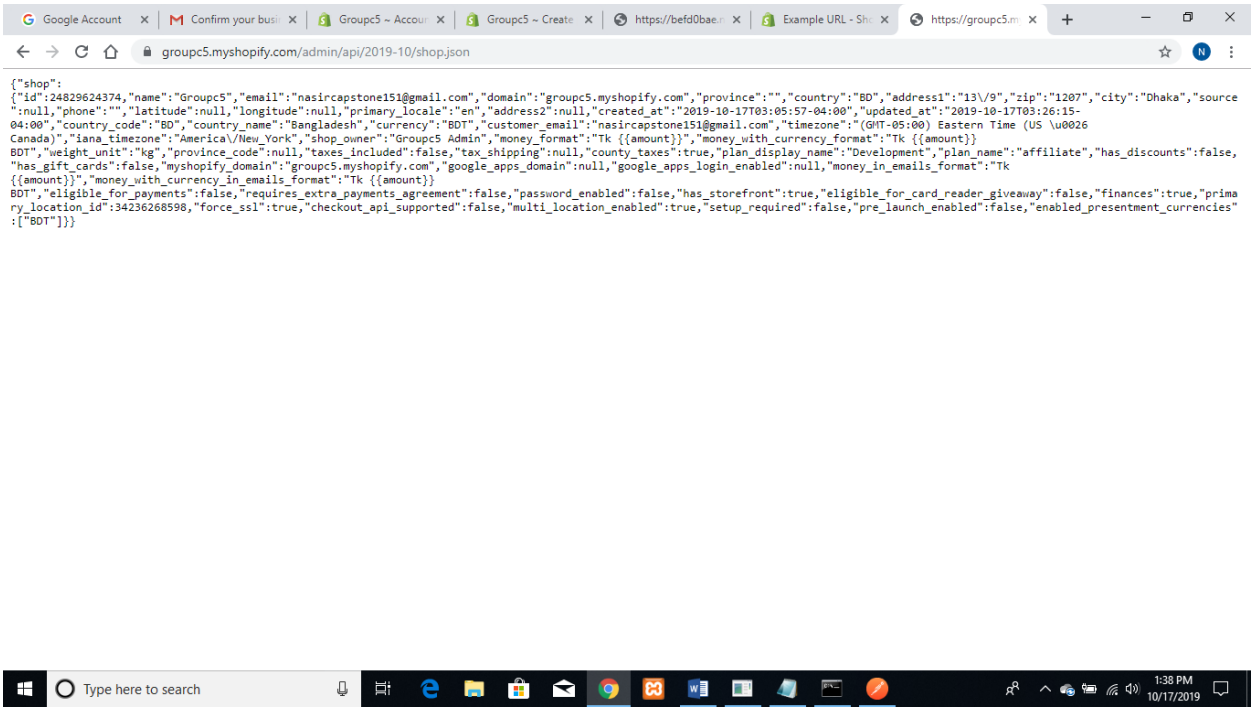
From here, Example URL is important. Copy that.

Go to Postman and put the example URL.
https://9ec6d2641ec2fdf1a38c953a5871d731:0df50e4703f93b44f6720128f4f9da16@groupc5.myshopify.com/admin/api/2019-10/shop.json.



This will show the api calling.

Go to your browser and put the example URL, you can GET the shop information.



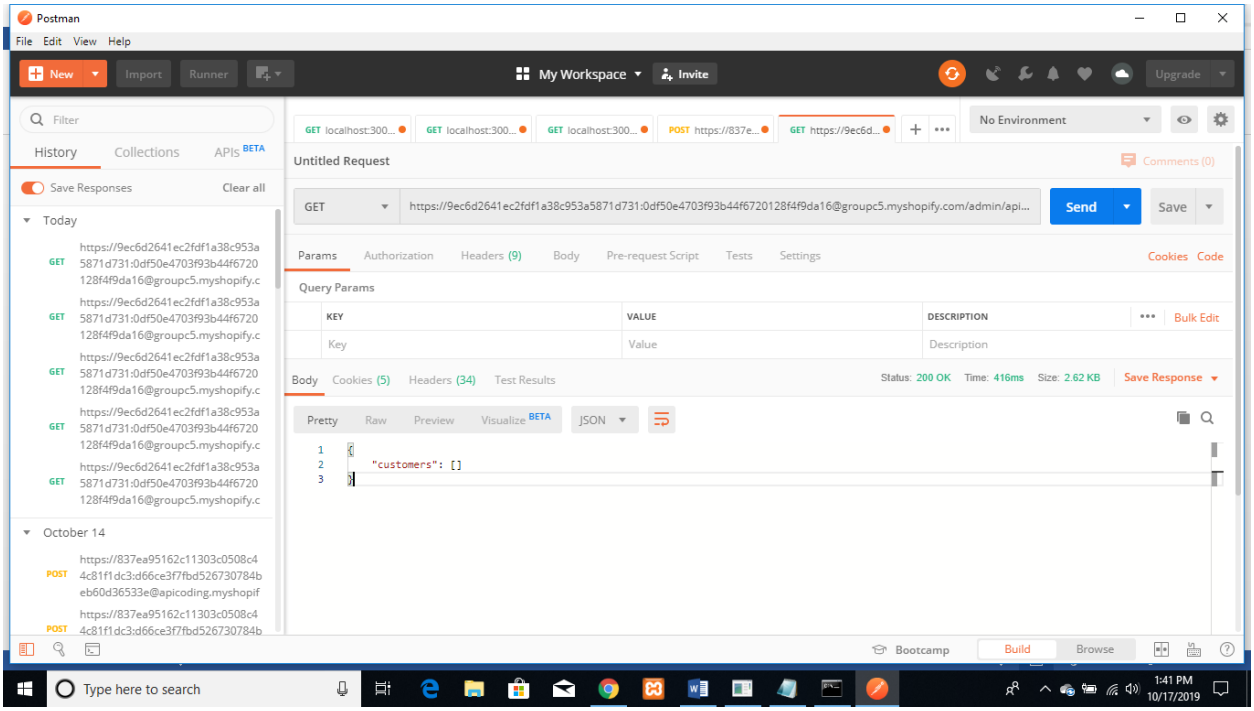
For registering an user.

Use the example URL in postman.

https://9ec6d2641ec2fdf1a38c953a5871d731:0df50e4703f93b44f6720128f4f9da16@groupc5.myshopify.com/admin/api/2019-10/customers.json

Note /api/2019-10/ after the slash, just put what API you need.

Initially there is no customer.



Put
https://9ec6d2641ec2fdf1a38c953a5871d731:0df50e4703f93b44f6720128f4f9da16@groupc5.myshopify.com/admin/api/2019-10/customers.json

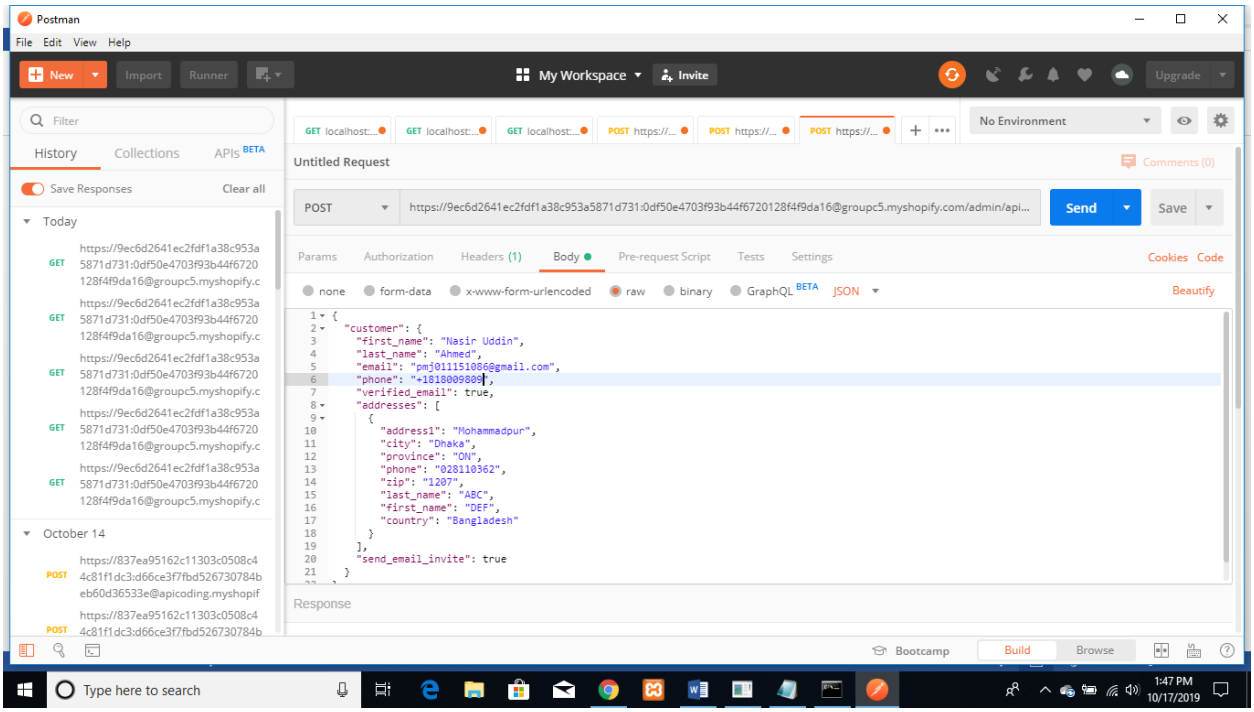
On postman, select Post.

Now copy the following request body in the postman.

```
{
  "customer": {
    "first_name": "Nasir Uddin",
    "last_name": "Ahmed",
    "email": "pmj011151086@gmail.com",
    "phone": "+15142546011",
    "verified_email": true,
    "addresses": [
      {
        "address1": "Mohammadpur",
        "city": "Dhaka",
```

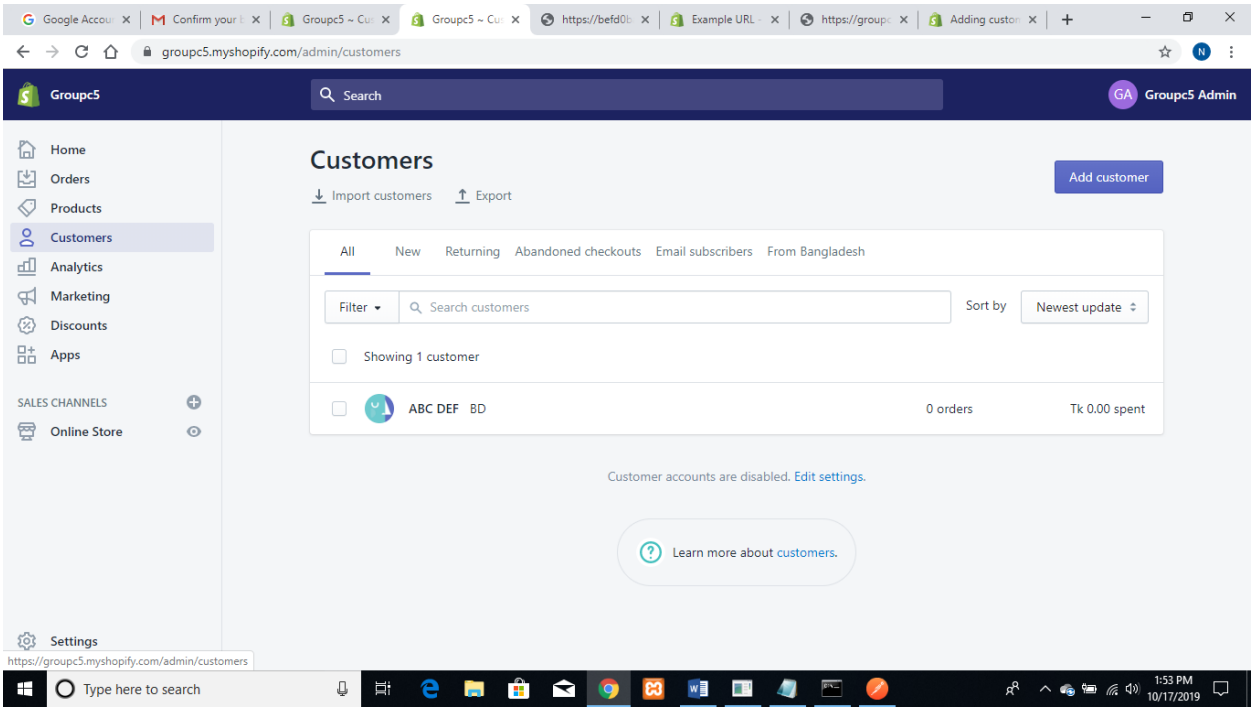
```
        "province": "ON",
        "phone": "028110362",
        "zip": "1207",
        "last_name": "ABC",
        "first_name": "DEF",
        "country": "Bangladesh"
      }
    ],
    "send_email_invite": true
  }
}
```

This code is available on https://community.shopify.com/c/Shopify-APIs-SDKs/Adding-customer-using-postman/td-p/345019

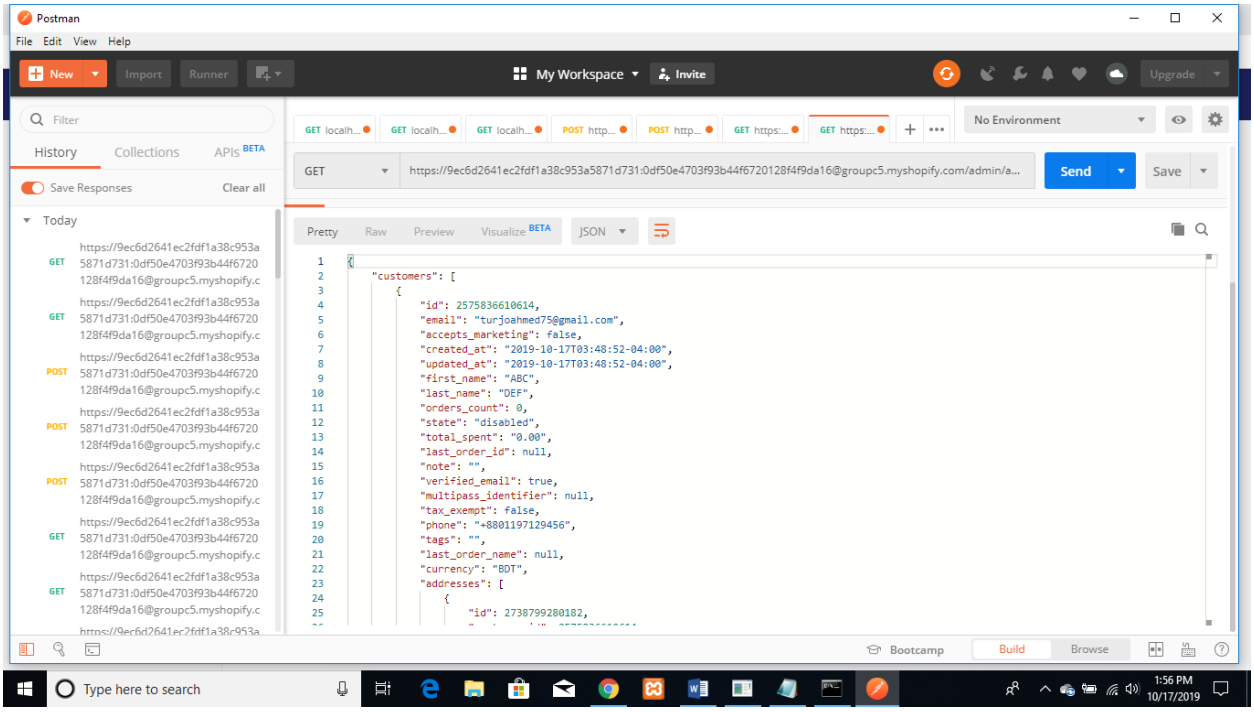


Then press send

Customer would be added to the Shopify store by api calling.



Using GET request and customers API, we can get registered customer.



Code link of the project => https://github.com/Turjo7/Integration-of-Multiple-Services-using-Application-Programming-Interface-API-

Thank You